Unclassified
Security Classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification: Unclassified		1b Restrictive Markings	
2a Security Classification Authority		3 Distribution Availability of Report: Approved for public release; distribution is unlimited.	
2b Declassification/Downgrading Schedule			
4 Performing Organization Report Number(s)		5 Monitoring Organization Report Number(s)	
6a Name of Performing Organization: Naval Postgraduate School	6b Office Symbol (*If Applicable*): 32	7a Name of Monitoring Organization: Naval Postgraduate School	
6c Address (*city, state, and ZIP code*): Monterey, CA 93943-5000		7b Address (*city, state, and ZIP code*): Monterey, CA 93943-5000	
8a Name of Funding/Sponsoring Organization	8b Office Symbol (*If Applicable*)	9 Procurement Instrument Identification Number	
8c Address (*city, state, and ZIP code*)		10 Source of Funding Numbers: Program Element Number / Project No / Task No / Work Unit Accession No	

11 Title (*Include Security Classification*): THE DESIGN OF AN ADAPTIVE ATTITUDE CONTROL SYSTEM

12 Personal Author(s): Nicholas F. Russo

13a Type of Report	13b Time Covered	14 Date of Report (*year, month,day*)	15 Page Count
Master's Thesis	From To	September 1992	91

16 Supplementary Notation: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes		
Field	Group	Subgroup

18 Subject Terms (*continue on reverse if necessary and identify by block number*): CER, attitude control, attitude hold, slewing, Space Station, eigenaxis, quaternion, Euler Parameters, Linear Quadratic Regulator, Kalman Filter

19 Abstract (*continue on reverse if necessary and identify by block number*)

This research designed and simulated an adaptive attitude control system for the Crew Equipment/Retriever (*CER*) during autonomous attitude hold and large angle or slewing maneuvers. The CER is a proposed space robot that deploys from the Space Station and retrieves any lost equipment or incapacitated astronauts. The moment of inertia tensor for the CER and acquired target is not known *a priori*. In this research, the moment of inertia tensor is estimated by a Kalman filter and used to update the derived linear quadratic regulator (*LQR*) and quaternion feedback regulator (*QFR*) control laws. Computer simulation results show that during attitude hold the adaptive LQR design stabilizes the CER and provides a more fuel efficient controller effort: as compared with a previously designed nonadaptive minimum time controller and a nonadaptive LQR design. Computer simulation results of slewing maneuvers show that the adaptive QFR design provides a more fuel efficient controller: as compared with a nonadaptive QFR design.

20 Distribution/Availability of Abstract	21 Abstract Security Classification	
[X] unclassified/unlimited [ ] same as report [ ] DTIC users	Unclassified	
22a Name of Responsible Individual: Jeffrey B. Burl	22b Telephone (*Include Area code*): (408) 646-2390	22c Office Symbol: EC/Bl

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted — All other editions are obsolete

security classification of this page: Unclassified

Approved for public release; distribution is unlimited.

The Design of an Adaptive Attitude Control System

by

Nicholas F. Russo
Lieutenant, United States Coast Guard
B.S., United States Coast Guard Academy, 1985

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1992

ABSTRACT

This research designed and simulated an adaptive attitude control system for the Crew Equipment/Retriever (*CER*) during autonomous attitude hold and large angle or slewing maneuvers. The CER is a proposed space robot that deploys from the Space Station and retrieves any lost equipment or incapacitated astronauts. The moment of inertia tensor for the CER and acquired target is not known *a priori*. In this research, the moment of inertia tensor is estimated by a Kalman filter and used to update the derived linear quadratic regulator (*LQR*) and quaternion feedback regulator (*QFR*) control laws. Computer simulation results show that during attitude hold the adaptive LQR design stabilizes the CER and provides a more fuel efficient controller effort: as compared with a previously designed nonadaptive minimum time controller and a nonadaptive LQR design. Computer simulation results of slewing maneuvers show that the adaptive QFR design provides a more fuel efficient controller: as compared with a nonadaptive QFR design.

TABLE OF CONTENTS

ACKNOWLEDGMENTS

I would like to thank Professor John Junkins (Texas A&M) for his knowledgeable support of this research and invaluable practical guidance. I give special thanks to Professor Jeff Burl for his patience and instructive guidance as my thesis advisor. I would also like to thank Professor Roberto Cristi for his guidance in adaptive control theory, filtering techniques, and his significant role in writing the journal article based on this research. To Lanette, Alex, and Tasha, thank you for all your love and support. I dedicate this thesis to you.

I. INTRODUCTION

A. CERS CONCEPT DESCRIPTION

1. CERS Origin and Purpose

The National Aeronautics and Space Administration (NASA), Johnson Space Center Space Station Projects Office sent out a *Request for Proposal* in May 1987 as part of Space Station Work Package 2. This *Request for Proposal*, including an added Amendment 7, defined a requirement to provide for the capability to rescue an incapacitated external-vehicular activity crewman and to retrieve equipment inadvertently detached from the Space Station. [Ref.1: p. L-2-14a]

McDonnell Douglas Astronautics Company (MDAC) responded to this *Request for Proposal* in September 1987 with a practical, low cost retriever concept. This concept was referred to as the Crew/Equipment Retrieval System (CERS). [Ref. 2: p. 1] This overall system consisted of a crew and equipment retriever vehicle (CER) and other Space Station based support systems.

The overall mission of the CER is to: deploy from the Space Station, acquire and capture the designated target, and return to the Space Station. A summary of the CERS capabilities, as defined by MDAC, is listed as follows: [Ref. 2: p. 9]

1. Retrieve an 850 pound target (includes 10% safety margin);
2. Total Deployment time of 120 minutes;
3. Retriever activated and deployed without assistance from an external-vehicular activity crewman;
4. Retriever senses own attitude, range, and range to target;

5. Retriever can be remotely operated from the Space Station;
6. Retriever accommodates a worst-case separation speed of 3.5 ft/sec;
7. Retriever senses and controls its own attitude with and without a target;
8. Retriever has attitude hold and three-axis translation capability.

2. CERS Baseline Configuration

Figure 1 shows the CER and its Space Station support systems. More detailed descriptions of proposed hardware and software are contained in Ref. 2 (pp. 20-68).

A simple representation of the CER for attitude dynamics analysis was developed in Ref. 3 and is shown in Figure 2. The characteristics of the baseline configuration developed by MDAC are listed as follows: [Ref. 2: p. 24]

1. 850 pounds total weight;
2. Three-axis (six degrees of freedom) stabilized;
3. Remote tele-operated free flyer;
4. Use of 24 cold Nitrogen (N_2) jet thrusters rated at 1.0 lbf each;
5. Attitude control is accomplished by firing thrusters in pairs;
6. Maximum control torques:

   Roll Axis 3 ft-lbf;

   Pitch Axis 3 ft-lbf;

   Yaw Axis 4 ft-lbf.

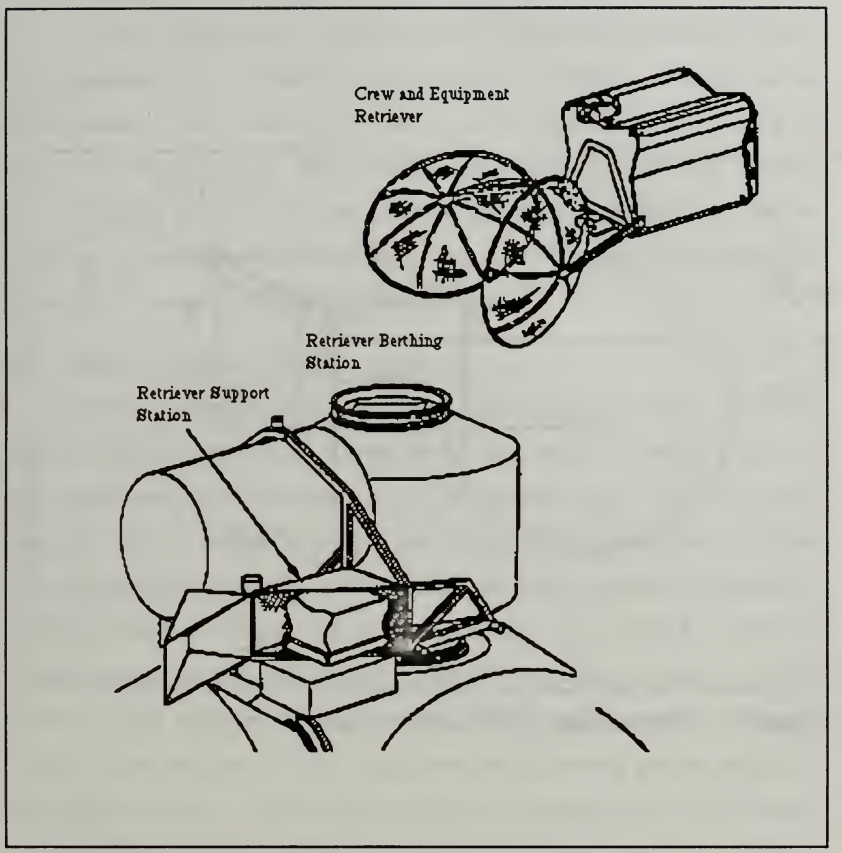

Figure 1. CERS Major Components [Ref. 2: p. 15]

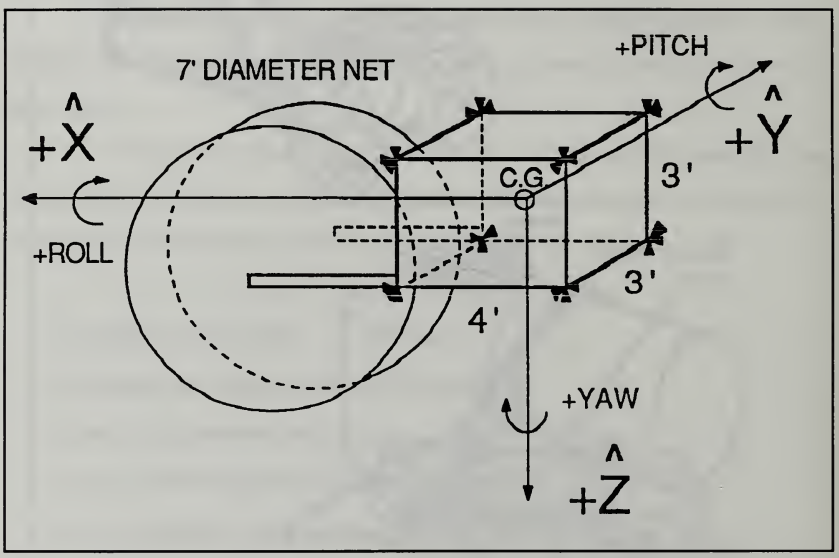

Figure 2. CER Baseline Configuration [Ref. 3: p. 8]

An important feature of the CER is its ability to control its attitude. After acquiring a target the CER must stabilize its attitude and perform an attitude reorientation prior to returning to the Space Station.

The CER's attitude control problem is very different from the norm. Most spacecraft are designed such that any changes in their moments of inertia are minimized. The control devices are, moreover, placed so as to act along the principal axes of the body. This positioning minimizes any gyroscopic coupling between torque applied to any one axis and rotation about another axis. Neither of these two conditions hold true for the CER after it acquires a target since the target can be as massive as the CER by itself.

B. THESIS OBJECTIVES

The overall objective of this thesis is to design adaptive attitude control laws for the CER with and without a target during large angle or *slewing maneuvers* and during autonomous *attitude hold* for all mission phases. Previous thesis research [Ref. 3] modeled the CER and designed time-optimal and weighted time-fuel optimal single-axis control systems and tested these control systems by applying them to several worst case target scenarios. This research analyzes the complete nonlinear three-axis control problem for small angle motion or attitude hold and large angle or slewing motion. Attitude hold pertains to attitude control in the presence of small disturbances while slewing motion pertains to attitude reorientation. The control law designs are adaptive in that a key system parameter, the moment of inertia of the CER and acquired target, is not known *a priori* and must be estimated. A subsidiary goal of the slewing motion control law is to accomplish this reorientation in an optimal fashion: an *eigenaxis* rotation. Both control laws must be able to deal with a non-diagonal moment of

inertia tensor since after target acquisition, the thrusters no longer act along the principal axes of the body.

C. THESIS ORGANIZATION

In Chapter II, general spacecraft attitude kinematics and dynamics are developed. These equations of motion are then applied to the CER. The moment of inertia tensors for the CER with and without a target are calculated.

Chapter III derives two control law designs. One design is developed for attitude hold while another design is developed for large angle or slewing maneuvers. Central to each control law design is the knowledge of the CER moment of inertia tensor.

In Chapter IV, an estimation scheme is developed that provides the above mentioned control laws with an estimate of the CER moment of inertia tensor. This estimation scheme is based on a rather unusual application of the Kalman Filter.

Chapter V presents the computer simulation results of applying each control law and estimation scheme. Control system design issues and practical implementation details are also discussed.

Conclusions based on the computer simulation results are presented in Chapter VI. In addition, recommendations for future research are discussed.

II. ATTITUDE KINEMATICS AND DYNAMICS

The equations of motion for any rotating rigid body can be divided into two sets: the *Kinematic Equations of Motion* and the *Dynamic Equations of Motion. Kinematics* studies motion without considering the forces that cause that motion. The *Kinematic Equations of Motion* are a set of first order differential equations that specify the time evolution of the chosen attitude parameters. The *Dynamic Equations of Motion*, meanwhile, take into account the forces that cause rotational motion and express the time evolution of the angular velocity of the rigid body. [Ref. 4: p. 510]

A. ROTATIONAL KINEMATICS

1. Direction Cosine Matrix

Any general vector $\underline{r}$ can be written in terms of its magnitude and direction. The direction can be represented as a unit vector referenced to some previously defined coordinate or reference frame as shown in Figure 3. This unit vector is made up of components known as *direction cosines*: [Ref. 5: p. 9]

$$\underline{r} = r\hat{r} = r[(\cos\alpha)\hat{n}_1 + (\cos\beta)\hat{n}_2 + (\cos\gamma)\hat{n}_3] \tag{2.1}$$

where r is a scalar magnitude, and $\hat{r}$ is a unit vector whose components are referenced to the three orthogonal axes of the reference frame $\hat{n}$. Note the following symbolic conventions used throughout this thesis:

1. The underline bar denotes a vector, $\underline{r}$
2. The hat symbol denotes a unit vector, $\hat{r}$;
3. The double underline bar denotes a matrix, $\underline{\underline{W}}$.

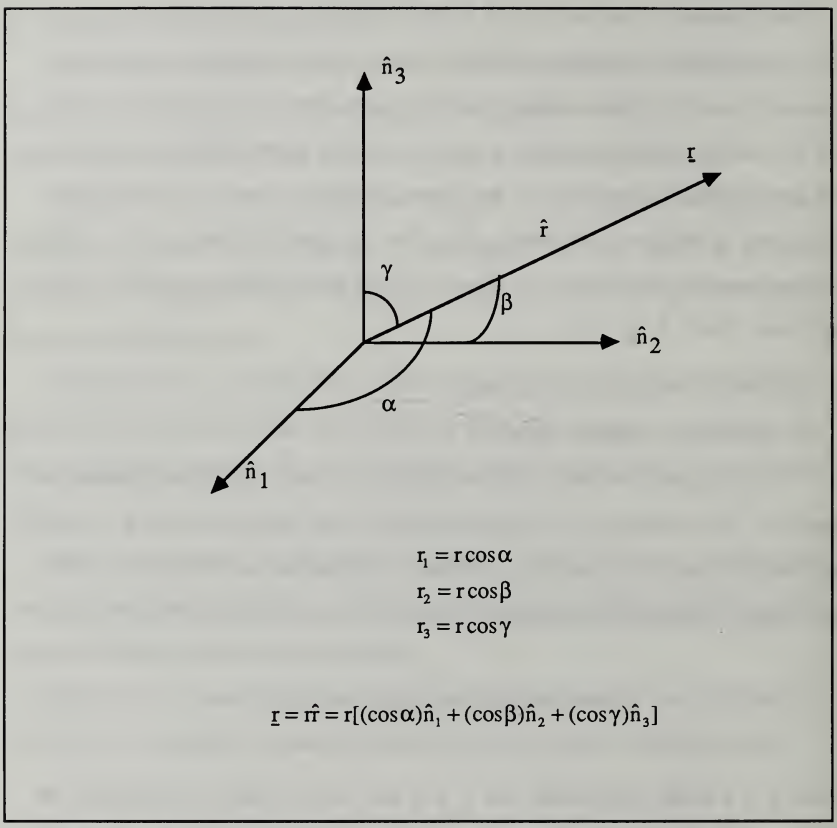

Figure 3. Direction Cosines [Ref. 5: p. 9]

Now consider a unit vector $\hat{b}$ with components along the orthogonal body axes of a rigid body, i.e., a spacecraft as shown in Figure 4. Let $\hat{n}$ be a unit vector with components along three orthogonal directions that are fixed in space. These two unit vectors are related by the following transformation: [Ref. 5: p. 9]

$$\hat{b} = \underline{\underline{T}}\hat{n} \; . \tag{2.2}$$

The transformation is defined by $\underline{\underline{T}}$, the 3x3 *direction cosines matrix* (DCM). This DCM is critical to the field of spacecraft dynamics and control and Ref. 5 addresses several of its important properties. The most important of these properties are summarized below:

1. A DCM exists for any pair of orthogonal sets of three axes;
2. The DCM is an orthogonal matrix and its inverse equals its transpose;
3. A DCM can be built up from successive rotations about the axes.

The last property is best explained by an example. Define a sequence of reference frames related by the following transformations:

$$\hat{a} = \underline{\underline{T}}_1\hat{n}; \tag{2.3}$$

$$\hat{b} = \underline{\underline{T}}_2\hat{a}; \tag{2.4}$$

$$\hat{c} = \underline{\underline{T}}_3\hat{b}; \tag{2.5}$$

and therefore the reference frames $\hat{c}$ and $\hat{n}$ are related by:

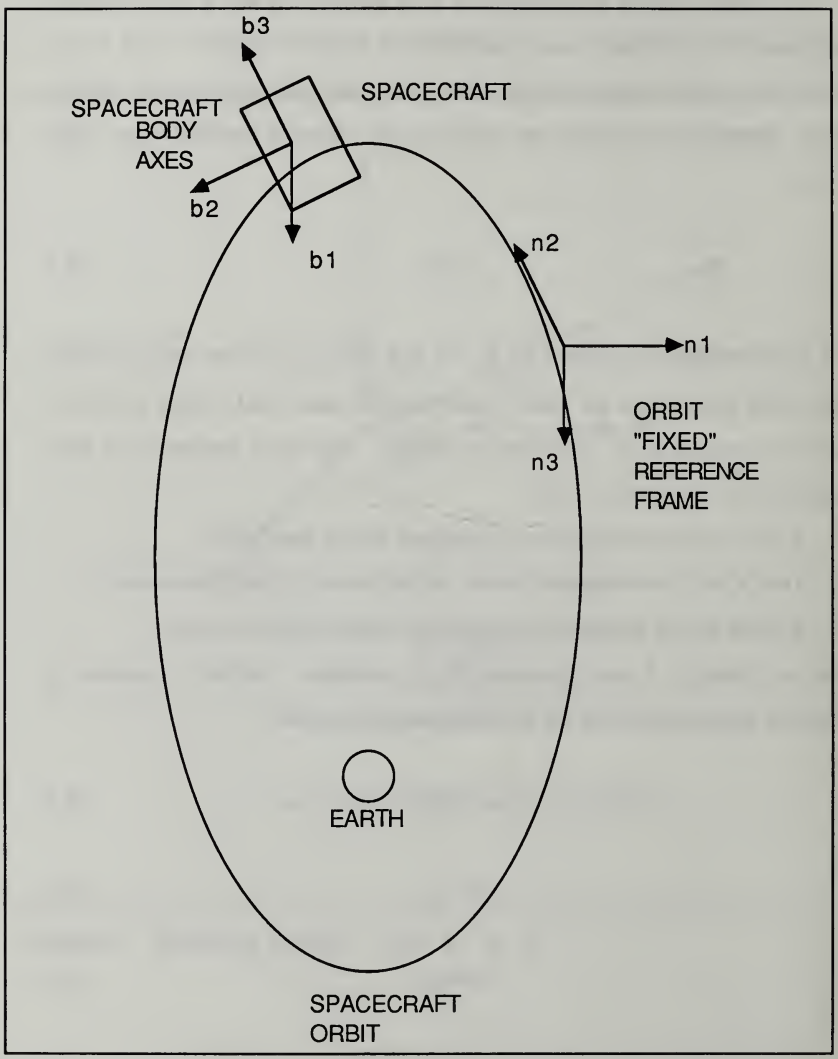

Figure 4. Spacecraft Body Axes

$$\hat{c} = \underline{\underline{T}}_3 \underline{\underline{T}}_2 \underline{\underline{T}}_1 \hat{n}. \tag{2.6}$$

In other words, one overall DCM can be formed as a product of DCMs:

$$\underline{\underline{T}}_{321} = \underline{\underline{T}}_3 \underline{\underline{T}}_2 \underline{\underline{T}}_1. \tag{2.7}$$

2. Euler Angles

The relative orientation of two orthogonal reference frames can be defined in terms of three angles [Ref. 5: p. 16]. This idea was first introduced by Euler in the early eighteenth century and is synonymous with the idea of parameterizing the previously discussed nine element DCM with only three independent parameters.

The classical Euler angles, for which there are twelve distinct cases, define an arbitrary orientation by using the successive rotational transformation property of the DCM. That is, a sequence of three elementary rigid right handed rotations about instantaneously fixed axes is used to build up an overall DCM that represents the transformation from one orientation to a different orientation. [Ref. 5: p. 17]

This thesis employs the *3-2-1* or *yaw-pitch-roll* Euler angle sequence since it is most commonly used in aircraft and spacecraft applications. This sequence is produced by initially lining up both the body axes and the fixed or inertial axes. A rotation about the *z* or *number three axis* is then performed, producing a new *y* and *x axis* . A second rotation about the new *y* or *number two axis* is then enacted. Finally, a rotation about the new *x axis* finishes the rotation sequence. Figure 5 depicts the formulation of this sequence.

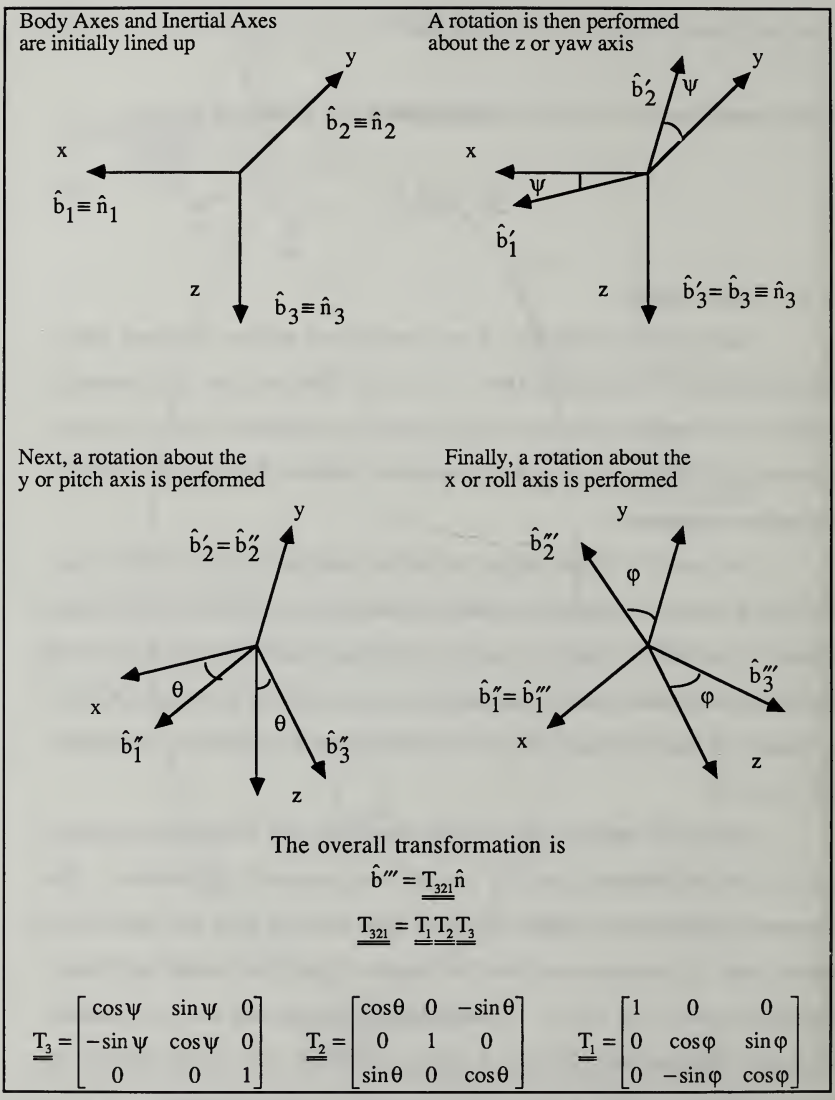

Figure 5. The 3-2-1 Euler Angles

Reference 5 illustrates the general process for obtaining the kinematic differential equations for a chosen set of Euler angles. Wertz [Ref. 4: p. 765] lists the kinematic equations of motion for the twelve possible Euler angle representations. The kinematic differential equations for the chosen *3-2-1* Euler angle set are as follows:

$$\dot{\psi} = (\omega_Y \sin(\varphi) + \omega_Z \cos(\varphi)) / \cos(\theta); \tag{2.8}$$

$$\dot{\theta} = \omega_Y \cos(\varphi) - \omega_Z \sin(\varphi); \tag{2.9}$$

$$\dot{\varphi} = \omega_X + (\omega_Y \sin(\varphi) + \omega_Z \cos(\varphi)) \tan(\theta); \tag{2.10}$$

where:

1. ψ is the yaw angle;
2. θ is the pitch angle;
3. φ is the roll angle;
4. The vector $\underline{\omega}$ is the angular velocity vector and is composed of components along each of the body axes.

3. Euler's Principal Rotation Theorem

Junkins [Ref. 5: p. 26] states that Euler is generally credited with being responsible for the *Principal Rotation Theorem*:

> A rigid body can be brought from an arbitrary initial orientation to an arbitrary final orientation by a single rotation of the body through a principle angle about a principal line; the principal line being a judicious axis fixed in the body and fixed in space.

This concept, displayed in Figure 6, allows the DCM to be parameterized in terms of the principal angle ϕ and principal line $\hat{\ell}$. In mathematical terms, the principal line corresponds to the eigenvector of the DCM: for the eigenvalue ± 1. Therefore, given any DCM one can solve for the principal line and angle and reduce the general angular displacement to a single rotation about a fixed line. [Ref. 5: p. 27]

4. Euler Parameters

In conjunction with the *Principal Rotation Theorem*, Euler defined four parameters in terms of the principal line and principal angle. These *Euler Parameters* are as follows:

$$\begin{aligned} \beta_0 &= \cos(\phi/2) \\ \beta_1 &= \ell_1 \sin(\phi/2) \\ \beta_2 &= \ell_2 \sin(\phi/2) \\ \beta_3 &= \ell_3 \sin(\phi/2) \end{aligned} \tag{2.11}$$

where ℓ_1, ℓ_2, ℓ_3 are the components of the unit vector along the principal line $\hat{\ell}$ and ϕ is the principal angle.

The DCM is therefore parameterized in terms of the above *Euler Parameters* and this allows the relative orientation of two orthogonal reference frames to be represented by four parameters. One of these parameters is redundant since the DCM was previously shown to be parameterized by three Euler angles. This redundancy manifests itself in the following constraint equation:

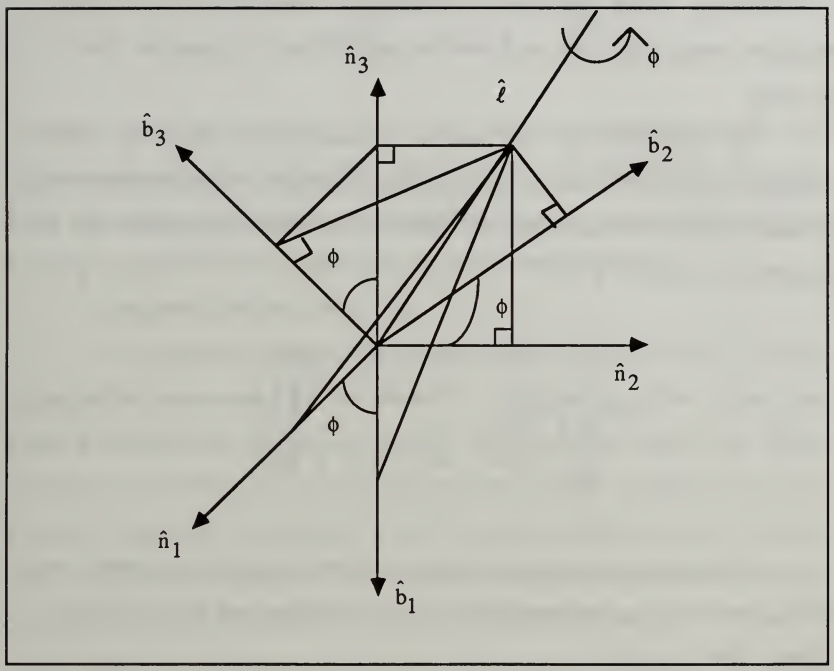

Figure 6. Euler's Principal Rotation Theorem [Ref. 5: p. 25]

$$\beta_0^2 + \beta_1^2 + \beta_2^2 + \beta_3^2 = 1. \tag{2.12}$$

Various algorithms have been developed that determine the Euler parameters from a given DCM and convert back and forth between Euler angles and parameters. These algorithms have been used extensively in the computer simulation programs for this thesis and are included in the Appendix. [Ref. 5: pp. 31-35]

By differentiating the inverse relationships between the Euler parameters and the elements of the DCM, and making a few substitutions, the kinematic differential equations in terms of the Euler parameters can be formulated as: [Ref. 5: p. 35]

$$\underline{\dot{\beta}} = \begin{bmatrix} \dot{\beta}_0 \\ \dot{\beta}_1 \\ \dot{\beta}_2 \\ \dot{\beta}_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}. \tag{2.13}$$

Note that throughout the literature on attitude dynamics and control, the Euler parameters are sometimes referred to as *quaternions* and are formulated as follows: [Ref. 4: p. 414]

$$\begin{aligned} q_1 &= \ell_1 \sin(\phi/2) \\ q_2 &= \ell_2 \sin(\phi/2) \\ q_3 &= \ell_3 \sin(\phi/2) \\ q_4 &= \cos(\phi/2) \end{aligned} \tag{2.14}$$

The constraint equation (2.12) is also applicable and a conversion from one representation of Euler parameters to this quaternion representation allows the following identification:

$$q_4 \equiv \beta_0 \qquad (2.15)$$
$$q_i \equiv \beta_i, \qquad i = 1, 2, 3.$$

As seen above in equation (2.15), the difference between these two representations is rather insignificant. Note that this author has chosen to use the Euler parameter representation of equation (2.11) but throughout this thesis the terms *Euler parameters* and *quaternions* are used interchangeably.

5. Parameterization Discussion

The previous sections briefly demonstrate that there exist several choices when representing the orientation of a rigid body. Euler angles are easier to visualize and are more popular but suffer from one large draw back: the presence of mathematical singularities at certain angles. Equation (2.8), for example, experiences a singularity when the cosine of the pitch angle goes to zero. This can be a real problem in terms of numerical computations during computer simulation or software running control system algorithms. The use of Euler parameters eliminates the use of trigonometric functions and their singularities but they are harder to visualize.

Therefore, the intended application should dictate the choice of kinematic differential equations. In this thesis, the *3-2-1* Euler angle set is used to define the kinematics for small angle motion during attitude hold. For large

angle or slewing motion and control, the less widely used but much more practical Euler parameters are used to represent the kinematics.

B. ROTATIONAL DYNAMICS

1. General Rigid Body

The Eulerian Rotational Equations of Motion for a rigid body subject to applied control torques are well known and are typically represented as: [Ref. 5: pp. 49-52]

$$\dot{\underline{\omega}} = -\underline{\underline{I}}^{-1}\underline{\underline{\tilde{\omega}}}\underline{\underline{I}}\underline{\omega} + \underline{\underline{I}}^{-1}\underline{u} \tag{2.16}$$

where:

1. $\dot{\underline{\omega}}$ is the angular acceleration vector;
2. $\underline{\underline{I}}$ is the moment of inertia tensor for the rigid body;
3. $\underline{\omega}$ is the angular velocity vector;
4. $\underline{u}$ is the vector of applied control torques;
5. $\underline{\underline{\tilde{\omega}}}$ is the skew symmetric angular velocity matrix defined as:

$$\underline{\underline{\tilde{\omega}}} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ \omega_2 & \omega_1 & 0 \end{bmatrix}. \tag{2.17}$$

2. The CER

The dynamics of the CER can be represented by equation (2.16) provided that the rigid body assumption is used. The only system parameter in equation (2.16) that is specific to the CER is its moment of inertia tensor. This moment of inertia tensor will change whenever the CER captures a target.

Hansen [Ref. 3] calculated the moment of inertia for the CER without target by assuming an 850 pound total system weight symmetrically distributed about the center of gravity. The resulting matrix is diagonal in the CER's body coordinate system and has the units of slug-foot^2:

$$\underline{\underline{I}}_{CER} = \begin{bmatrix} 39.6 & 0 & 0 \\ 0 & 55 & 0 \\ 0 & 0 & 55 \end{bmatrix}. \tag{2.18}$$

The total moment of inertia of the CER with a target, as previously discussed, is not known *a priori* since it depends on the specific mass and moment of inertia of the acquired target. Hansen [Ref. 3: p. 17] utilized the parallel axis theorem to calculate the total moment of inertia for the CER and an acquired target

$$\underline{\underline{I}}_{TOTAL} = \underline{\underline{I}}_{CER} + \underline{\underline{I}}_{TARGET} - \frac{M_1 M_2}{M_1 + M_2} \underline{\underline{R}}_2^{\,2} \tag{2.19}$$

where M_1 is the mass of the CER, M_2 is the mass of the target, and $\underline{\underline{R}}_2$ is the skew symmetric matrix derived from the vector $\underline{r}_2$ between the center of gravity of the CER and the target:

$$\underline{\underline{R}}_2 = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix}. \tag{2.20}$$

The fact that vectors can only be added together if they are expressed in the same coordinate system also applies to the moments of inertia in equation (2.19). Here

again, the DCM $\underline{\underline{T}}$ plays the role of transforming a moment of inertia from one reference frame to another reference frame as follows:

$$\underline{\underline{I}}^{b}_{TARGET} = \underline{\underline{T}}\,\underline{\underline{I}}^{n}_{TARGET}\,\underline{\underline{T}}^{T} \tag{2.21}$$

where the superscript *T* refers to taking the transpose of the given matrix, the superscripts *b* and *n* refer to the reference frame, and $\underline{\underline{I}}$ is the moment of inertia tensor.

Hansen [Ref. 3: pp. 17-21] further utilized equations (2.18-2.21) to calculate five *worst case* target scenarios, each with an associated total moment of inertia tensor. This thesis used the *Case Two* scenario most frequently for control system design analysis and computer simulation. This scenario corresponds to an astronaut with manned maneuvering unit in the target net and has a total moment of inertia (slug-foot^2) of:

$$\underline{\underline{I}}_{TOTAL} = \begin{bmatrix} 112.9 & 2.4 & -111.9 \\ 2.4 & 534.9 & 6.4 \\ -111.9 & 6.4 & 497.6 \end{bmatrix}. \tag{2.22}$$

C. STATE VARIABLE REPRESENTATION

The total set of equations representing the CER attitude kinematics and dynamics is composed of equation (2.16) and either equations (2.8-2.10) for the *3-2-1* Euler angle representation or equation (2.13) for the Euler parameter representation. These equations are nonlinear in nature but for initial control system design and as an approximation when small angle motion is considered, equation (2.16) and equations (2.8-2.10) can be approximated as:

$$\dot{\underline{X}} = \underline{\underline{A}}\underline{X} + \underline{\underline{B}}\underline{u} \tag{2.23}$$

where $\underline{X} = \begin{bmatrix} \psi & \theta & \varphi & \omega_z & \omega_y & \omega_x \end{bmatrix}^T$, $\underline{u}$ is the applied control torques and:

$$\underline{\underline{B}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ I_{11}^{-1} & I_{12}^{-1} & I_{13}^{-1} \\ I_{12}^{-1} & I_{22}^{-1} & I_{23}^{-1} \\ I_{13}^{-1} & I_{23}^{-1} & I_{33}^{-1} \end{bmatrix}; \tag{2.24}$$

$$\underline{\underline{A}} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{2.25}$$

In order to complete the state space equations, the *plant* defined by equations (2.23-2.25) must be accompanied by an output equation:

$$\underline{Y} = \underline{\underline{C}}\underline{X} + \underline{\underline{D}}\underline{u}. \tag{2.26}$$

This thesis assumes that all the states are measurable and there is no direct coupling between control input and the state outputs. Therefore, the following definitions are made:

$$\underline{\underline{C}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}; \tag{2.27}$$

$$\underline{\underline{D}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.28}$$

Note that the $\underline{\underline{C}}$ matrix is simply the identity matrix and the $\underline{\underline{D}}$ matrix is simply a matrix of zeros. Therefore, equation (2.26) can be reduced to:

$$\underline{Y} = \underline{X}. \tag{2.27}$$

Realistically, equation (2.27) should contain some added errors or noise due to the fact that sensors have limited accuracy and do introduce noise into any actual system. A more detailed discussion on sensors is contained in Chapter III and a related discussion on computer simulation implementation details is found in Chapter V.

III. ATTITUDE CONTROL LAW DESIGN

The development of the attitude equations of motion in the previous chapter and the design of control laws or algorithms in this chapter are based on one fundamental assumption: attitude motion of a spacecraft can be approximately decoupled from its orbital motion. For the purpose of attitude control design, therefore, the spacecraft is almost universally considered to have only rotational degrees of freedom about its center of mass which is fixed to a reference frame moving on the orbital path. In reality, attitude and orbital dynamics are coupled and environmental torques produced by gravity gradient, aerodynamic, and solar radiation pressure depend on the spacecraft's orientation. [Ref. 6: p. 7]

A. DESIGN PHILOSOPHY

The goal of control system design is to cause the output variable of some dynamic system or process to follow a desired reference variable accurately in spite of changes in this reference variable, the external disturbances applied, and any changes in the dynamics of the process itself. Prior to beginning any control system design, a mathematical model of the system to be controlled is constructed. [Ref. 7: p. 17] The dynamic process to be controlled in this thesis is the attitude of the CER plus any acquired target and its equations of motion have been developed in the previous chapter.

The process of *regulation* defines a situation in which the output variable of some dynamic process must follow a constant, usually zero, reference variable. [Ref. 7: p. 107]. The attitude control of the CER can be defined as a regulation process. While maneuvering or after acquiring a target, the CER's attitude must

remain constant despite the presence of external disturbance forces. After target acquisition, the CER must perform a reorientation before initiating a translational maneuver that will return it to the Space Station. In this reorientation maneuver, the reference variable is the desired orientation and the output variable is the current orientation. A regulation process is needed to reduce the orientation error, which represents the difference between the current and desired orientation, to zero in a timely fashion.

This thesis designs *closed-loop* or *feedback* control systems because of their inherent ability to reject disturbances and errors in the model of the dynamic process to be controlled. This decision, by definition, requires the introduction of an output sensor which can introduce noise into the control system [Ref. 7: pp. 107-113]. The availability of quality sensors to measure angular position and angular velocity is assumed in this research. Wertz [Ref. 4: pp. 155-201] describes in detail the various types of hardware available to determine a spacecraft's angular position and angular velocity. The most common instruments used are the rate gyro and rate-integrating gyro. Since both instruments have a long history of operation, and are relatively inexpensive, the assumption that quality sensors exist is, therefore, very reasonable.

B. SPECIFICATIONS AND CONTROL DEVICES

The typical feedback control system is designed such that the overall dynamic system response meets a set of predetermined specifications. These specifications, more often than not, must be translated into terms more readily understood by the control engineer.

One set of specifications is known as *time domain specifications* and include such information as settling time, maximum overshoot, and damping ratio. In

terms of the CER's dynamics, this refers to how long it takes for a reorientation maneuver and how much orientation angle overshoot is allowed before settling down to the final desired value. Typical spacecraft attitude reorientation control requires that no overshoot occurs and this corresponds to a damping ratio of 1.0. Reference 1 does not specifically address the issue of settling time for CER attitude reorientation maneuvers or attitude regulation in the presence of disturbances. However, MDAC [Ref. 2: p. 9] defines a total deployment of time of 120 minutes. This is certainly an upper limit for any reorientation maneuvers. A more realistic figure is obtained from MDAC's [Ref. 2: p. 17] definition of major mission phases as a function of time. In this mission phase sequence, ten minutes is allocated to target capture. In this thesis, 70 seconds was selected as a reasonable settling time to accomplish any reorientation maneuver. Any attitude regulation, after the CER is subject to a disturbance, must be accomplished in a fraction of this time. A more detailed discussion of specifications and performance will be given in the following sections on attitude hold and slewing maneuvers.

In order to achieve control over the CER some type of control device must be selected. Wertz [Ref. 4: p. 201-210] discusses in detail the types of devices available for spacecraft control. The most widely used type of device are gas jets or thrusters and these are the control devices that MDAC selected in their baseline design of the CER.

The choice of the control actuators is, furthermore, tied to the desired specifications and to what is practically available. MDAC, for example, has chosen 1 lbf cold nitrogen thrusters. Wertz [Ref. 4: p. 206] mentions that gas jets are classified as *cold gas* and *hot gas*. Hot gas jets typically produce high

thrust levels (>5 N or 1.12 lbf) but rely upon a chemical reaction which must reach steady state. Cold gas systems produce lower thrust levels ($\leq$ 1 N or 0.225 lbf) but do not rely upon a chemical reaction which must reach steady state. Cold gas systems, therefore, provide more precise control and can be used effectively in a pulsed mode. This thesis assumes the use of the cold gas thrusters selected by MDAC. These thrusters are commercially available and can operate in a pulsed mode.

C. ATTITUDE HOLD

1. Optimal Control Theory

A linear feedback control law is defined in the following form:

$$\underline{u} = -\underline{\underline{K}}\underline{X} \tag{3.1}$$

where:

1. $\underline{u}$ is the applied control effort;
2. $\underline{\underline{K}}$ is a gain matrix that must be determined;
3. $\underline{X}$ is the vector containing all the state variables, assuming that they are all available by either measurement or estimation.

The gain matrix $\underline{\underline{K}}$ can be determined by choosing appropriate Laplace domain closed-loop pole locations based on some given time domain specifications. This method, known as *pole placement*, only works well for single-input-single-output (SISO) systems. In multi-input-multi-output (MIMO) systems, such as the CER, this technique does not lead to the development of a unique control law. In *optimal control theory*, the gain matrix $\underline{\underline{K}}$ is determined by minimizing a specified performance criterion or *cost function.* [Ref. 8: pp. 337-338]

2. Linear Quadratic Regulator

The *linear quadratic regulator* is an optimal control law of the form shown in equation (3.1). It is called *linear* because the control law is a linear function of the system states and it is called a *regulator* because this type of control law is well suited for regulation type problems. The *quadratic* description relates to the fact that the gain matrix $\underline{\underline{K}}$, of equation (3.1), is calculated by minimizing a quadratic integral cost function. For the continuous state space system described by equation (2.23), a quadratic integral cost function can be formulated as:

$$J = \int_{t_0}^{t_f} [\underline{X}^T(\tau)\underline{\underline{Q}}\underline{X}(\tau) + \underline{u}^T(\tau)\underline{\underline{R}}\underline{u}(\tau)]d\tau \tag{3.2}$$

where $\underline{\underline{Q}}$ and $\underline{\underline{R}}$ are symmetric weighting matrices that must be chosen by the control system designer. $\underline{\underline{Q}}$ penalizes deviation of the state vector $\underline{X}$ from the origin and $\underline{\underline{R}}$ penalizes the use of too much control effort. [Ref. 8: pp. 339-340]

During attitude hold of the CER, the goal is to reject all disturbances and maintain a constant attitude with zero angular velocity. If all the states are considered initialized at zero, then the control effort must drive all states towards the origin after a disturbance causes a deviation from this situation. A typical value for $\underline{\underline{Q}}$ that will cause the position angles to go to the origin and yet limit the angular velocities is

$$\underline{\underline{Q}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & 0 & c & 0 \\ 0 & 0 & 0 & 0 & 0 & c \end{bmatrix} \tag{3.3}$$

where c is a number less than one [Ref. 8: p. 340].

The choice of a proper control weighting matrix $\underline{\underline{R}}$ also requires careful consideration. Unless a cost or penalty is imposed for using too much control, the design that emerges may generate control signals that can not be achieved by the actuators or control devices. The resulting control signal then saturates at the maximum signal value that can be produced and this produces, in most cases, the fastest possible response. The fastest possible response may be highly desirable but the closed-loop behavior of a system in saturation may be quite different from the closed loop behavior predicted by a system not in saturation. The system may even become unstable when the control system saturates and because of this consequence, $\underline{\underline{R}}$ should be chosen to avoid saturation. [Ref. 8: p. 341]

Hansen [Ref. 3] designed a control law based on saturating the CER's control input. This control scheme produced a *minimum time* solution but it required the control thrusters to be either on or off; this is known in the literature as *bang-bang control.* Some of the *worst case target* scenarios produced unstable results and this caused Hansen to choose larger values and different locations for the CER thrusters in order to avoid an unstable situation.

The gain matrix $\underline{\underline{K}}$, that minimizes equation (3.2), is found by solving the *Riccati Equation.* It is, in general, a time varying matrix that, given fairly

general conditions which apply to the CER, eventually reaches a steady state value. This optimum gain matrix in the steady state is given by:

$$\underline{\underline{K}}_{ss} = \underline{\underline{R}}^{-1}\underline{\underline{B}}^{T}\overline{\underline{\underline{M}}} \tag{3.4}$$

where $\overline{\underline{\underline{M}}}$ satisfies the algebraic equation also known as the *algebraic Riccati equation*:

$$\underline{0} = \overline{\underline{\underline{M}}}\underline{\underline{A}} + \underline{\underline{A}}^{T}\overline{\underline{\underline{M}}} - \overline{\underline{\underline{M}}}\underline{\underline{B}}\underline{\underline{R}}^{-1}\underline{\underline{B}}^{T}\overline{\underline{\underline{M}}} + \underline{\underline{Q}}. \tag{3.5}$$

[Ref. 8: pp. 345-346] Many software packages, including the program MATLAB used in this thesis, contain subroutines that calculate the steady state value of $\underline{\underline{K}}$ for a given dynamic system and cost function.

Attitude hold for the CER is, therefore, accomplished by using a Linear Quadratic Regulator to drive the states of the system to the origin after experiencing some external disturbances. The gain matrix $\underline{\underline{K}}$ is determined by supplying a MATLAB subroutine with a model of the CER's dynamics and appropriately chosen $\underline{\underline{Q}}$ and $\underline{\underline{R}}$ weighting matrices. The steady state value of $\underline{\underline{K}}$ is used for simplicity since $\underline{\underline{K}}$ only varies near the final time and this situation is not encountered during most of the CER's mission.

D. SLEWING MANEUVERS

The problem of reorienting a spacecraft from one rest orientation to another rest orientation, although a problem of *regulation*, requires the formulation of a quite different control law. The linearized equations of motion, used for attitude hold design, are no longer valid. Slewing over a potentially large range of

orientation angles requires the use of the complete nonlinear equations of motion defined by equations (2.13) and (2.16). A linear control law may not be adequate for this task and the formulation of a nonlinear control law may be required. In addition, any slewing maneuver should, ideally, be an *optimal* maneuver. *Optimal* in this context refers to taking the shortest angular path.

1. Eigenaxis Rotations

Many spacecraft attitude control systems are currently based on a sequence of rotational maneuvers about each control axis. This is a natural bias based on the popularity of Euler angles for describing rigid body orientation. However, the maneuver time of such successive rotations is two to three times longer than that of a single maneuver about the *eigenaxis*. This *eigenaxis* is the principal axis developed in Euler's Principal Rotation Theorem. Euler, moreover, proved that the principal angle ϕ of equation (2.11) is always smaller than the algebraic sum of three successive Euler angles and represents the shortest angular path between two relative orientations. Therefore, a control law that causes a spacecraft to reorient itself by rotating about the *eigenaxis* will be executing an *optimal* maneuver. [Ref. 9: pp. 375-376]

2. Quaternion Feedback Regulator

Reference 9 develops a nonlinear control law that takes into account the complete nonlinear attitude equations of motion and executes an *eigenaxis* rotational maneuver. This development begins by defining a *quaternion error* which represents the attitude error between the current orientation and the desired or commanded orientation:

$$\begin{bmatrix} \beta_{0e} \\ \beta_{1e} \\ \beta_{2e} \\ \beta_{3e} \end{bmatrix} = \begin{bmatrix} \beta_{0c} & \beta_{1c} & \beta_{2c} & \beta_{3c} \\ -\beta_{1c} & \beta_{0c} & \beta_{3c} & -\beta_{2c} \\ -\beta_{2c} & -\beta_{3c} & \beta_{0c} & \beta_{1c} \\ -\beta_{3c} & \beta_{2c} & -\beta_{1c} & \beta_{0c} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} \tag{3.6}$$

where terms in the matrix represent the commanded orientation expressed in Euler parameters. A three-dimensional error vector can be formed by extracting the three vector components of equation (3.6) as:

$$\underline{\beta}_e = \begin{bmatrix} \beta_{1e} \\ \beta_{2e} \\ \beta_{3e} \end{bmatrix}. \tag{3.7}$$

The complete quaternion regulator feedback control law is:

$$\underline{u} = \underline{\tilde{\omega}}\underline{I}\underline{\omega} - \underline{D}\underline{\omega} - \underline{K}\underline{\beta}_e \tag{3.8}$$

where the first term is a nonlinear body-rate feedback term that counteracts the gyroscopic coupling torque found in equation (2.16), the second term is a linear body-rate feedback term, the third term is a linear error-quaternion or Euler parameter-error feedback term, and $\underline{D}$ and $\underline{K}$ are 3x3 constant gain matrices to be properly determined. [Ref. 9: p. 376]

To complete the control law in equation (3.8), the matrices $\underline{D}$ and $\underline{K}$ must be determined. Reference 9 considered the gain selections

$$\underline{K} = k\underline{I}; \tag{3.9}$$

$$\underline{\underline{D}} = d\underline{\underline{I}} \tag{3.10}$$

where k and d are positive scalars, and $\underline{\underline{I}}$ is the spacecraft moment of inertia tensor. Global stability via Lyapunov stability analysis was proved for the control law of equation (3.8) provided that

$$\underline{\underline{K}}^{-1}\underline{\underline{D}} > 0. \tag{3.11}$$

Equation (3.11) is always guaranteed with the gains defined by equations (3.9-3.10).

With global stability guaranteed for the control law in equation (3.8), all that remains is a proper choice of k and d. Let $\hat{\lambda}$ be a unit vector along the eigenaxis. The orientation is then expressed as:

$$\underline{\beta} = \sin(\phi/2)\hat{\lambda}. \tag{3.12}$$

Substituting equation (3.8) into equation (2.16) yields the following closed-loop equation:

$$\dot{\underline{\omega}} = -\underline{\underline{I}}^{-1}\tilde{\underline{\omega}}\underline{\underline{I}}\underline{\omega} + \underline{\underline{I}}^{-1}\left(\tilde{\underline{\omega}}\underline{\underline{I}}\underline{\omega} - \underline{\underline{D}}\underline{\omega} - \underline{\underline{K}}\underline{\beta}\right). \tag{3.13}$$

Further substitution of equations (3.9-3.10) produces:

$$\dot{\underline{\omega}} = \underline{\underline{I}}^{-1}\left(-d\underline{\underline{I}}\underline{\omega} - k\underline{\underline{I}}\underline{\beta}\right) \tag{3.14}$$

or by rearranging terms:

$$\underline{\underline{I}}\dot{\underline{\omega}} + d\underline{\underline{I}}\underline{\omega} + k\underline{\underline{I}}\underline{\beta} = 0 \tag{3.15}$$

When the angular rate $\underline{\omega}$ is small, and when an eigenaxis rotation is assumed, the angular rate can be approximated as:

$$\omega = \dot{\phi}\hat{\lambda}. \tag{3.16}$$

Further substitution of equations (3.16) and (3.12) into equation (3.15) yields:

$$\left(\ddot{\phi} + d\dot{\phi} + k\sin(\phi/2)\right)\underline{\underline{I}}\hat{\lambda} = 0. \tag{3.17}$$

Since the moment of inertia $\underline{\underline{I}}$ is, by definition, a positive definite matrix and the unit vector $\hat{\lambda}$ is non zero, then $\underline{\underline{I}}\hat{\lambda} \neq 0$ and equation (3.17) becomes:

$$\ddot{\phi} + d\dot{\phi} + k\sin(\phi/2) = 0. \tag{3.18}$$

If the $\sin(\phi/2)$ is approximated by $\phi/2$, equation (3.18) is further reduced to:

$$\ddot{\phi} + d\dot{\phi} + k\phi/2 = 0. \tag{3.19}$$

Equation (3.19) is the well-known linear second order equation where the damping ratio ζ and the natural frequency ω_N satisfy:

$$d = 2\zeta\omega_N; \tag{3.20}$$

$$k = 2\omega_N^2. \tag{3.21}$$

Therefore, proper selection of the damping ratio and the natural frequency defines the positive scalars d and k. [Ref. 9: p. 377-378]

Since this thesis has assumed a damping ratio of one, a required maneuver *settling time* is converted to a required natural frequency by:

$$T_s = 4/\zeta\omega_N. \tag{3.22}$$

To account for the nonlinear effect of $\sin(\phi/2)$ when ϕ is large, equation (3.22) was modified by Ref. 9 as:

$$T_s = 8/\zeta\omega_N. \tag{3.23}$$

Therefore, a quaternion feedback regulator control law is defined by a desired orientation and by a desired maneuver settling time as:

$$d = 2\omega_N = 16/T_s; \tag{3.24}$$

$$k = 2\omega_N^2 = 128/T_s^2; \tag{3.25}$$

$$\underline{u} = \underline{\tilde{\omega}}\underline{\underline{I}}\underline{\omega} - (16/T_s)\underline{\underline{I}}\underline{\omega} - (128/T_s^2)\underline{\underline{I}}\underline{\beta}_e. \tag{3.26}$$

IV. ADAPTATION LAW DESIGN

Knowledge of the moment of inertia tensor for the CER and acquired target is required by both the control laws developed in Chapter III. In the case of the large angle control law, imprecise knowledge of the moment of inertia tensor results in a *non-eigenaxis* slewing maneuver. An argument can also be made that precise knowledge of the moment of inertia tensor should limit the amount of thruster firings required and save propellant fuel. Therefore, the ability to estimate the moment of inertia tensor and provide this information to the control algorithms is a very beneficial addition to the previously developed control algorithms.

A. ADAPTIVE CONTROL THEORY

An *adaptive* controller differs from a *static* or ordinary controller in only one respect. In an adaptive controller, the controller parameters are variable and there is a mechanism for adjusting these parameters *on-line* based on signals available from the overall system. The two main approaches for constructing adaptive controllers are: the *model reference method* and the *self-tuning method.* A schematic representation of each of these methods is presented in Figure 7. [Ref. 10: p. 315]

This thesis employs the *self-tuning method.* The design of an adaptive controller by the self-tuning method involves choosing a control law based on variable parameters and choosing an adaptation law for adjusting those parameters. The controller, therefore, couples a previously designed control law with an *on-line parameter estimator.* [Ref. 10: pp. 319-323] The previous

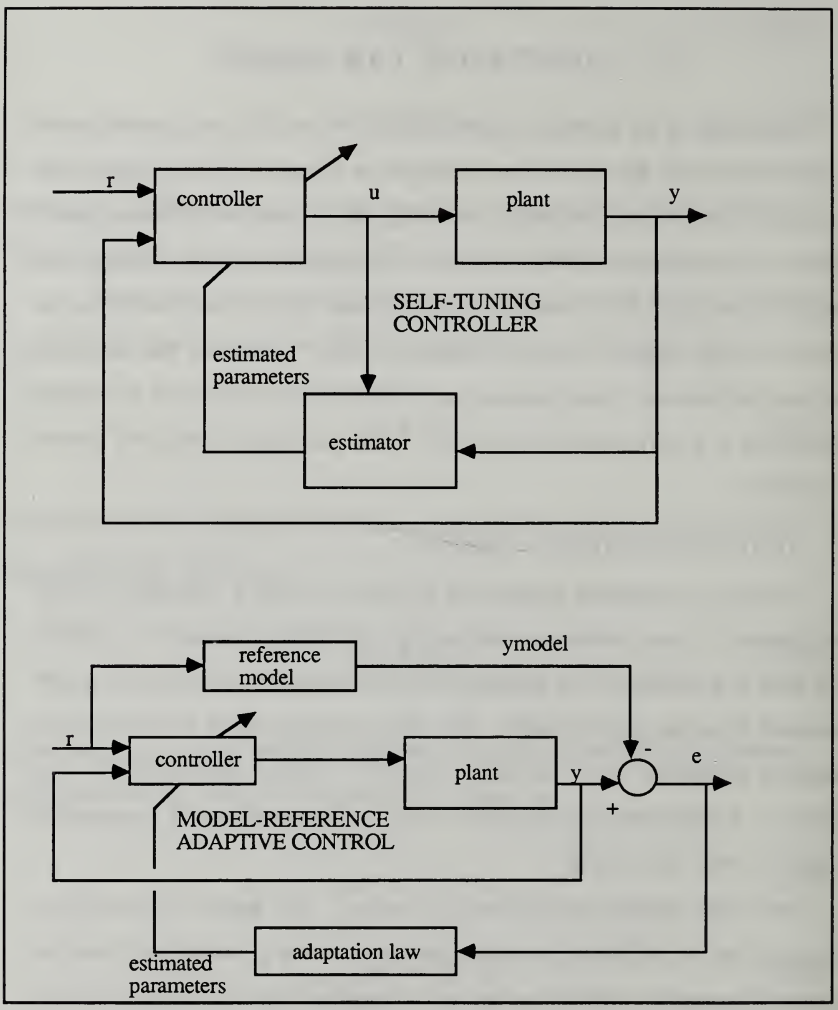

Figure 7. Adaptive Control Methods [Ref. 10: pp. 315, 320]

chapter developed two control laws based on knowledge of the CER moment of inertia tensor; the next section develops an adaptation law based on the Kalman filter equations.

B. KALMAN FILTER DESIGN

1. General Kalman Filter Equations

The Kalman filter is an observer or state estimator for a dynamic process given by the following discrete state space equation:

$$\underline{X}(k+1) = \underline{\underline{\Phi}}\underline{X}(k) + \underline{\underline{\Delta}}_1\underline{u}(k) + \underline{\underline{\Delta}}_2\underline{w}(k) \tag{4.1}$$

where:

1. $\underline{X}(k)$ is the state vector at the present time k;
2. $\underline{X}(k+1)$ is the state vector one time step in the future;
3. $\underline{\underline{\Phi}}$ is the discrete-time version of the *A* matrix given in equation (2.23);
4. $\underline{u}(k)$ is the applied control;
5. $\underline{w}(k)$ is an unknown random input called the *plant driving noise*;
6. $\underline{\underline{\Delta}}_1$ is the discrete version of the *B* matrix given in equation (2.23);
7. $\underline{\underline{\Delta}}_2$ is the random input influence matrix and is often identical to $\underline{\underline{\Delta}}_1$.

In addition, the measurements of the system are given by:

$$\underline{Y}(k) = \underline{\underline{C}}\underline{X}(k) + \underline{v}(k) \tag{4.2}$$

where $\underline{v}(k)$ is a random vector known as *measurement noise.* [Ref. 11: p. 27]

The Kalman filter provides an optimal estimate of the system described by equation (4.1) and (4.2) by minimizing the mean square error between the actual states and the estimated states. The Kalman filter has the following form:

$$\underline{\hat{X}}(k+1|k+1) = \underline{\hat{X}}(k+1|k) + \underline{\underline{G}}(k+1)\left[\underline{Y}(k+1) - \underline{\hat{Y}}(k+1|k)\right] \tag{4.3}$$

where:

1. the notation $(k+1|k)$ refers to the discrete value at time $k+1$ based on data accumulated through time k;
2. $\underline{\hat{X}}(k+1|k)$ is the estimate of the states given data through time k;
3. $\underline{\hat{Y}}(k+1|k)$ is the estimate of the measurements given data through time k;
4. $\underline{\underline{G}}(k+1)$ is known as the Kalman filter gain;
5. $\underline{\hat{X}}(k+1|k+1)$ is estimate of the states given data through time $k+1$.

A set of recursive equations, in the proper order, that solve for the Kalman filter gain and equation (4.3) are:

$$\underline{\underline{P}}(k+1|k) = \underline{\underline{\Phi}}\,\underline{\underline{P}}(k|k)\underline{\underline{\Phi}}^T + \underline{\underline{\Delta_2}}\,\underline{\underline{W}}\,\underline{\underline{\Delta_2}}^T; \tag{4.4}$$

$$\underline{\underline{G}}(k+1) = \underline{\underline{P}}(k+1|k)\underline{\underline{C}}^T\left[\underline{\underline{C}}\,\underline{\underline{P}}(k+1|k)\underline{\underline{C}}^T + \underline{\underline{V}}\right]^{-1}; \tag{4.5}$$

$$\underline{\underline{P}}(k+1|k+1) = \left[\underline{\underline{1}} - \underline{\underline{G}}(k+1)\underline{\underline{C}}\right]\underline{\underline{P}}(k+1|k); \tag{4.6}$$

$$\underline{\hat{X}}(k+1|k) = \underline{\underline{\Phi}}\,\underline{\hat{X}}(k|k) + \underline{\underline{\Delta_1}}\,\underline{u}(k); \tag{4.7}$$

$$\underline{\hat{Y}}(k+1|k) = \underline{\underline{C}}\,\underline{\hat{X}}(k+1|k); \tag{4.8}$$

$$\underline{\hat{X}}(k+1|k+1) = \underline{\hat{X}}(k+1|k) + \underline{\underline{G}}(k+1)\left[\underline{Y}(k+1) - \underline{\hat{Y}}(k+1|k)\right]; \qquad (4.9)$$

where:

1. $\underline{\underline{P}}$ is the covariance matrix of the estimation error;
2. $\underline{\underline{W}}$ is the covariance matrix of the plant driving noise;
3. $\underline{\underline{V}}$ is the covariance matrix of the measurement noise;
4. $\underline{\underline{1}}$ is an appropriately dimensioned identity matrix.

Note that final implementation of equations (4.4-4.9) requires an initial estimate of the states $\underline{\hat{X}}(0|0)$ and an initial estimate of the covariance matrix $\underline{\underline{P}}(0|0)$ [Ref. 11: pp. 27-29]. The initialization of equations (4.4-4.9) is discussed in Chapter V.

2. Linear Model

For attitude hold, the *linear quadratic regulator* requires a model of the CER's dynamics and this is provided by equations (2.23-2.25). The *B* matrix defined by equation (2.24) contains the elements of the CER's inverse moment of inertia tensor which is the system parameter that needs to be estimated.

To apply the previously defined Kalman filter equations, a plant equation similar to equation (4.1) is formulated as:

$$\underline{I}^{-1}(k+1) = \underline{I}^{-1}(k) + \underline{w}(k) \qquad (4.10)$$

where $\underline{I}^{-1}(k)$ is a vector that contains the six independent elements of the inverse moment of inertia tensor and $\underline{w}(k)$ is random plant driving noise. By comparing equations (4.1) and (4.10), the following correspondences are noted:

$$\underline{\underline{\Phi}} = \underline{\underline{1}}; \tag{4.11}$$

$$\underline{\underline{\Delta_1}} = \underline{\underline{0}}; \tag{4.12}$$

$$\underline{\underline{\Delta_2}} = \underline{\underline{1}}. \tag{4.13}$$

This basically assumes that the inverse moment of inertia does not change, other than by the addition of some random noise, from one time step until the next.

An accompanying measurement equation is formulated from a linearized version of equation (2.16) as

$$\dot{\underline{\omega}} = \underline{\underline{I}}^{-1}\underline{u}. \tag{4.14}$$

Equation (4.14) can be algebraically rearranged into the following form:

$$\dot{\underline{\omega}}(k) = \begin{bmatrix} u_1(k) & u_2(k) & u_3(k) & 0 & 0 & 0 \\ 0 & u_1(k) & 0 & u_2(k) & u_3(k) & 0 \\ 0 & 0 & u_1(k) & 0 & u_2(k) & u_3(k) \end{bmatrix} \underline{\underline{I}}^{-1}(k) + \underline{v}(k) \tag{4.15}$$

where $\underline{v}(k)$ is the combined measurement noise that results when angular acceleration and applied control torques are measured. By comparing equations (4.2) and (4.15), the following correspondence is noted:

$$\underline{\underline{C}}(k) = \begin{bmatrix} u_1(k) & u_2(k) & u_3(k) & 0 & 0 & 0 \\ 0 & u_1(k) & 0 & u_2(k) & u_3(k) & 0 \\ 0 & 0 & u_1(k) & 0 & u_2(k) & u_3(k) \end{bmatrix}. \tag{4.16}$$

The previously defined Kalman filter can now be used to estimate the inverse moment of inertia tensor of the CER plus any acquired target. This on-line estimation assumes that:

1. Reasonable angular acceleration measurements or estimates are available;
2. The applied control torques are available;
3. The covariance matrices for plant and measurement noise can be computed;
4. Initial estimates for the inverse moment of inertia and covariance matrix of estimation error can be computed.

All the above assumptions are discussed further in Chapter V.

3. Nonlinear Model

An estimate of the moment of inertia for the CER and any acquired target is required by the *quaternion feedback regulator* in order to accomplish any slewing maneuvers. A plant equation similar to equation (4.10) is formed as:

$$\underline{I}(k+1) = \underline{I}(k) + \underline{w}(k) \tag{4.17}$$

where equations (4.11-4.13) are still valid. Since large angle motion is inherently nonlinear, it is only prudent to form a measurement equation from the original nonlinear version of equation (2.16) which has been rearranged as:

$$\underline{u} = \underline{\tilde{\omega}}\,\underline{\underline{I}}\,\underline{\omega} + \underline{\underline{I}}\,\underline{\dot{\omega}}. \tag{4.18}$$

Further algebraic manipulation results in the following form:

$$\underline{u}(k) = \underline{\underline{C}}(k)\underline{I}(k) \tag{4.19}$$

where:

$$\underline{\underline{C}}(k) = \begin{bmatrix} \dot{\omega}_1 & \dot{\omega}_2 - \omega_1\omega_3 & \dot{\omega}_3 + \omega_1\omega_2 & -\omega_3\omega_2 & \omega_2^2 - \omega_3^2 & \omega_3\omega_2 \\ \omega_1\omega_3 & \dot{\omega}_1 + \omega_2\omega_3 & \omega_3^2 - \omega_1^2 & \dot{\omega}_2 & \dot{\omega}_3 - \omega_1\omega_2 & -\omega_1\omega_3 \\ -\omega_1\omega_2 & \omega_1^2 - \omega_2^2 & \dot{\omega}_1 - \omega_2\omega_3 & \omega_1\omega_2 & \dot{\omega}_2 + \omega_1\omega_3 & \dot{\omega}_3 \end{bmatrix} \quad (4.20)$$

and the time dependence of each term is assumed. Note that while the dynamics of the CER are nonlinear, the dynamics of the CER's moment of inertia remain linear.

The previously defined Kalman filter can now be used with this model to estimate the moment of inertia tensor of the CER plus any acquired target. This on-line estimation scheme utilizes the same assumptions listed in the previous section.

V. RESULTS

A. SIMULATION PROGRAM AND IMPLEMENTATION

Attitude control performance for the CER and target was evaluated through computer simulation. The simulation programs were written in MATLAB and are listed in the Appendix. The continuous-time equations of motion given in Chapter II and the continuous-time control algorithms given in Chapter III were simulated using discrete-time versions of these equations. A small sampling period of 75 milliseconds was chosen to ensure that the discrete equations accurately represented the continuous-time equations.

The selection of thruster size and location inherently limits the amount of available control torques. To ensure realistic simulation results, hard limits have been included in the simulation programs so that control signals greater than those given in Chapter I are not generated.

Plant noise has been added into the simulation programs to further increase the realism of the computer simulation results. Typical spacecraft disturbance torques are due to thruster misalignment and solar pressure. Kaplan [Ref. 12: p. 241] lists some assumed values for disturbance torques. The maximum magnitudes of these assumed values are:

1. Thruster misalignment torque: 8.5×10^{-5} N-m or 6.27×10^{-5} ft-lbf;
2. Solar Pressure torque: $\pm 1.0 \times 10^{-4}$ N-m or $\pm 7.4 \times 10^{-5}$ ft-lbf.

As a worst case guess, plant noise was modeled as a random three dimensional vector with a normal or Gaussian distribution and a standard deviation equal to the sum of these two disturbance torques, 1.4×10^{-4} ft-lbf.

The control laws developed in Chapter II assume the availability of sensors to measure angular position and angular velocity. The sensors have limited accuracy and introduce measurement noise into the control system. Wertz [Ref. 4: pp. 199-200] states that typical rate gyros have a resolution of less than 0.01 degree per second and typical rate-integrating gyros have a resolution of 0.003 degree. Reference 13 indicated that accuracies of 0.01 degree and 0.0003 degree per second are quite reasonable. As a worst case guess, measurement noise was modeled as two random three dimensional vectors with a normal or Gaussian distribution and standard deviations equal to 0.003 degree and 0.0001 degree per second.

The linear quadratic regulator used for attitude hold required a choice of $\underline{\underline{Q}}$ and $\underline{\underline{R}}$ weighting matrices. After some trial and error, weighting matrices were obtained that allowed the states to be driven towards the origin in a reasonable amount of time (ten seconds) and also avoided saturation of the applied control torques:

$$\underline{\underline{Q}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \tag{5.1}$$

$$\underline{\underline{R}} = \begin{bmatrix} 2\text{x}10^{-5} & 0 & 0 \\ 0 & 2\text{x}10^{-5} & 0 \\ 0 & 0 & 2\text{x}10^{-5} \end{bmatrix}. \tag{5.2}$$

Finally, the adaptation algorithm developed in Chapter IV required some computations and assumptions. Reference 13 indicated that angular accelerometers exist for spacecraft applications but are very expensive and not as common as rate gyros and rate-integrating gyros. In this thesis, angular acceleration measurements were generated by numerically differentiating the given angular velocity measurements as follows:

$$\dot{\underline{\omega}}(k) = (\underline{\omega}(k+1) - \underline{\omega}(k)) / T_s \tag{5.3}$$

where T_s is the sample period or interval. Alternative methods for generating angular acceleration measurements might involve filtering the already available angular velocity measurements or using an extended Kalman filter to estimate the moment of inertia tensor and the angular accelerations simultaneously. The initial covariance of estimation error matrix $\underline{\underline{P}}(0|0)$ was calculated empirically by computing the total moment of inertia tensor for several targets in various locations within the CER's net and then computing the overall mean and standard deviation. The covariance matrix of plant driving noise $\underline{\underline{W}}$ and the covariance matrix of measurement noise $\underline{\underline{V}}$ were computed as follows:

$$\underline{\underline{W}} = w\underline{\underline{1}}; \tag{5.4}$$

$$\underline{\underline{V}} = v\underline{\underline{1}}; \tag{5.5}$$

where $\underline{\underline{1}}$ is an appropriately dimensioned identity matrix, and v and w are scalar values equal to the variances of the assumed plant noise and measurement noise.

The plant is the CER and target moment of inertia and this author assumes that it does not change very significantly. A small number should, therefore, be chosen for w, with zero an ideal choice. However, simulations indicated that a zero plant noise covariance matrix leads to unsatisfactory results. In this thesis, the following small number was used:

$$w = 1x10^{-8}. \tag{5.6}$$

The measurement noise takes into account the errors and noise that are introduced by thruster misalignment, and measuring angular acceleration and angular velocity. After some trial and error via computer simulations, the value chosen empirically for v was:

$$v = 1x10^{-4}. \tag{5.7}$$

The adaptation scheme constantly updates the control algorithms as it estimates the moment of inertia. The Kalman filter has some inevitable transients prior to convergence and computer simulation results indicated that the Kalman filter may even provide an estimate for the moment of inertia that is not physically realistic. Since the moment of inertia tensor is by definition a positive definite matrix (the eigenvalues are always positive for a real physical body), a test has been incorporated within the control and adaptation schemes: estimates of the moment of inertia are only passed to the control law if all the eigenvalues are positive.

Hansen [Ref. 3: p. 21] computed the total moment of inertia tensor for the CER and several *worst case* target capture scenarios. These moments of inertia, tabulated in Table 1, are used in the following sections to evaluate the control laws developed in this thesis.

TABLE 1. TARGET CAPTURE MOMENT OF INERTIA TENSORS

Case/Description	Moment of Inertia Tensor (slug-foot^2)
1. No Target	$\begin{bmatrix} 39.6 & 0 & 0 \\ 0 & 55 & 0 \\ 0 & 0 & 55 \end{bmatrix}$
2. Man + manned maneuvering unit in net center not aligned with CER.	$\begin{bmatrix} 112.9 & 2.4 & -111.9 \\ 2.4 & 534.9 & 6.4 \\ -111.9 & 6.4 & 497.6 \end{bmatrix}$
3. 850 lb point mass at net edge along the x-axis.	$\begin{bmatrix} 69.3 & 0 & -178.2 \\ 0 & 1153.8 & 0 \\ -178.2 & 0 & 1124.1 \end{bmatrix}$
4. 850 lb point mass at net edge along the y-axis.	$\begin{bmatrix} 98.9 & -110.5 & -110.5 \\ -110.5 & 496.1 & -29.7 \\ -110.5 & -29.7 & 496.1 \end{bmatrix}$
5. 850 lb point mass at net edge along the z-axis.	$\begin{bmatrix} 369.5 & 0 & -368.4 \\ 0 & 796.4 & 0 \\ -368.4 & 0 & 466.4 \end{bmatrix}$
6. 850 lb point mass at X=Z, Y=1 from net center.	$\begin{bmatrix} 172.7 & -93.7 & -282.3 \\ -93.7 & 839.7 & -39.8 \\ -282.3 & -39.8 & 732.9 \end{bmatrix}$

B. ATTITUDE HOLD

Evaluation of attitude hold for the CER and acquired target was accomplished by initializing the position angles and angular velocities at some reasonable values and then allowing the control system to drive these values towards the origin. This thesis employed the same the initial conditions as Hansen [Ref. 3: p. 43]: 2.0 degrees for all position angles and 0.2 degrees per second for all angular velocities. Throughout the attitude hold simulations, the adaptive control system has been initialized with the moment of inertia defined by Case 1 in Table 1.

1. Comparison of Adaptive and Nonadaptive Control

The control system developed in Chapter III was compared with a nonadaptive version of the same control system for a CER and target moment of inertia defined by Case 2 in Table 1. The nonadaptive controller was given a constant value for the CER and target moment of inertia: defined by Case 1 in Table 1. The simulation results are listed below in Table 2.

TABLE 2. ADAPTIVE/NONADAPTIVE SIMULATION RESULTS

Controller Design	Settling Time (sec)	Fuel Used
1. Nonadaptive, I= Case 1	30	1009
2. Adaptive	10	398

A figure of merit for any spacecraft control system design is the amount of fuel used. In this thesis, the sum of the absolute value of all control used for each design was calculated. This value is listed as *fuel used* in Table 2, although it is

actually only proportional to the amount of fuel used. Note that the settling time in Table 2 is the maximum settling time for all three position angles.

Simulation plots for the nonadaptive design (I= Case 1) and the adaptive design are displayed in Figure 8. Both designs provide for an overall stable closed loop system. However, the nonadaptive design is clearly much more oscillatory, requires a long period for all oscillations to completely dampen out, and requires more control effort. The adaptive design starts out with the worst possible guess (the CER alone) for the moment of inertia but rapidly and correctly estimates the actual moment of inertia of the CER and target. The final estimate is very close to the actual moment of inertia defined by Case 2 in Table 1:

$$\underline{\underline{I}}_{est} = \begin{bmatrix} 112.7 & 2.8 & -111.5 \\ 2.8 & 540 & 3.3 \\ -111.5 & 3.3 & 498.1 \end{bmatrix}. \tag{5.8}$$

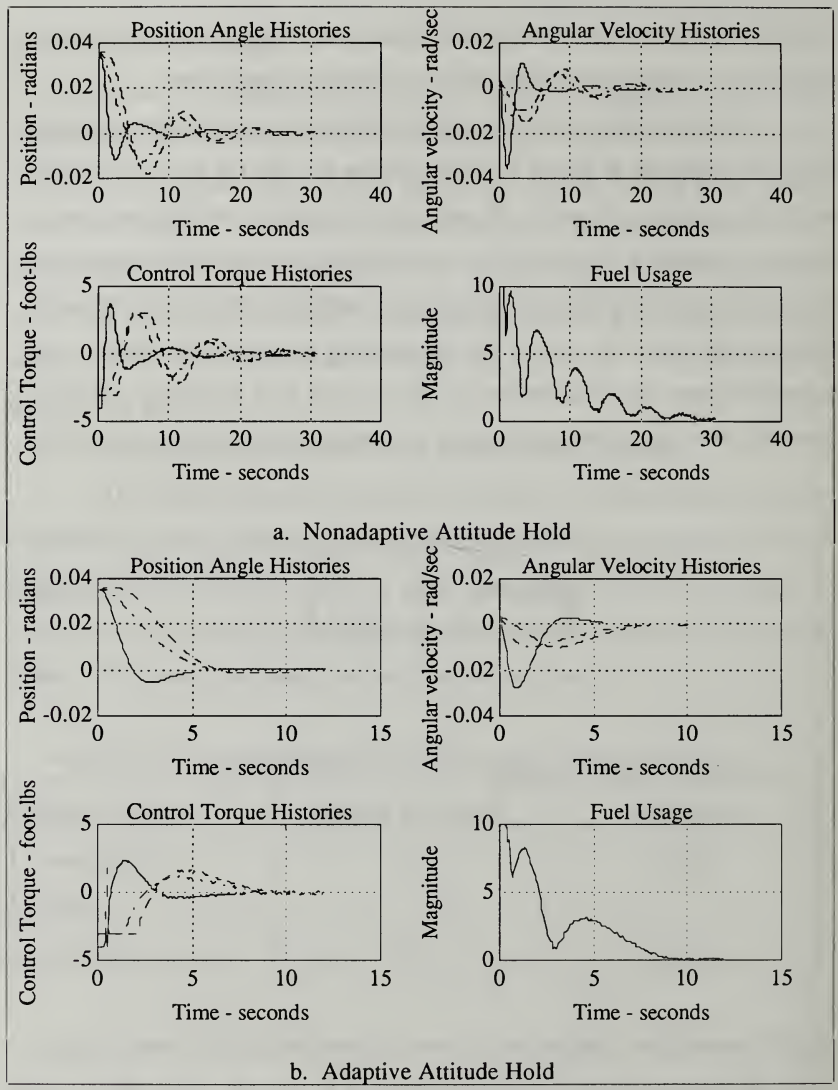

Figure 8. Simulation Results for Attitude Hold

2. Adaptive Control with various target scenarios

The adaptive control system developed for attitude hold in Chapter III was simulated for all the target-capture moment of inertia tensors defined by Hansen [Ref. 3]: Cases 1-6 in Table 1. The simulation results for this adaptive attitude control design and the nonadaptive minimum time controller designed by Hansen [Ref. 3] are listed below in Table 3.

TABLE 3. ATTITUDE HOLD SIMULATION RESULTS

Target Case	Settling Time (sec) Adaptive Design	Settling Time (sec) Hansen Design
1	3.0	1.5
2	10.0	6.6
3	15.0	∞
4	8.0	∞
5	15.0	15.6
6	13.0	∞

The adaptive control system was stable and exhibited reasonable settling times for all target cases. The minimum time control system designed by Hansen [Ref. 3] exhibited unstable results for three of the target capture cases: as depicted by the infinite settling times.

Table 4 lists the final estimate of the CER and target moment of inertia tensor for each of the target cases. The results are close but not exact. The differences may be attributed to the use of *noisy* measurement signals and the short time (10 seconds) available for system measurements. As the control

torque and angular acceleration measurements become very small, the estimation scheme is essentially provided with no new information and therefore can not accurately estimate the moment of inertia tensor.

TABLE 4. MOMENT OF INERTIA ESTIMATES

Target Case	Actual Moment of Inertia Tensor	Estimate of Moment of Inertia Tensor
1	$\begin{bmatrix} 39.6 & 0 & 0 \\ 0 & 55 & 0 \\ 0 & 0 & 55 \end{bmatrix}$	$\begin{bmatrix} 39.6 & -0.3 & 0.3 \\ -0.3 & 55.2 & 0.2 \\ 0.3 & 0.2 & 54.4 \end{bmatrix}$
2	$\begin{bmatrix} 112.9 & 2.4 & -111.9 \\ 2.4 & 534.9 & 6.4 \\ -111.9 & 6.4 & 497.6 \end{bmatrix}$	$\begin{bmatrix} 112.7 & 2.8 & -111.5 \\ 2.8 & 540 & 3.3 \\ -111.5 & 3.3 & 498.1 \end{bmatrix}$
3	$\begin{bmatrix} 69.3 & 0 & -178.2 \\ 0 & 1153.8 & 0 \\ -178.2 & 0 & 1124.1 \end{bmatrix}$	$\begin{bmatrix} 69.3 & -7.0 & -179.4 \\ -7.0 & 1155.8 & -4.4 \\ -179.4 & -4.4 & 1131.6 \end{bmatrix}$
4	$\begin{bmatrix} 98.9 & -110.5 & -110.5 \\ -110.5 & 496.1 & -29.7 \\ -110.5 & -29.7 & 496.1 \end{bmatrix}$	$\begin{bmatrix} 98.7 & -110.6 & -109.9 \\ -110.6 & 284.5 & 182.0 \\ -109.9 & 182.0 & 282.7 \end{bmatrix}$
5	$\begin{bmatrix} 369.5 & 0 & -368.4 \\ 0 & 796.4 & 0 \\ -368.4 & 0 & 466.4 \end{bmatrix}$	$\begin{bmatrix} 411.4 & 5.0 & -417.0 \\ 5.0 & 797.0 & -5.9 \\ -417.0 & -5.9 & 522.9 \end{bmatrix}$
6	$\begin{bmatrix} 172.7 & -93.7 & -282.3 \\ -93.7 & 839.7 & -39.8 \\ -282.3 & -39.8 & 732.9 \end{bmatrix}$	$\begin{bmatrix} 173.7 & -90.2 & -286.1 \\ -90.2 & 851.5 & -52.6 \\ -286.1 & -52.6 & 746.8 \end{bmatrix}$

C. SLEWING MANEUVERS

Evaluation of slewing maneuvers for the CER and acquired target was accomplished by: initializing the position angles and angular velocities at zero,

selecting a final orientation in terms of the chosen *3-2-1 Euler Angles*, and selecting a desired maneuver settling time. The adaptive quaternion feedback regulator developed in Chapters III and IV was compared with two nonadaptive quaternion feedback regulators that assumed a CER and target moment of inertia as defined by Case 1 in Table 1. The adaptive quaternion feedback regulator was initialized with this same moment of inertia. A desired orientation of 50 degrees for each position angle was selected for each controller design. The simulation results for a target defined as Case 2 are listed below in Table 5.

TABLE 5. SLEWING MANEUVER SIMULATION RESULTS

Controller	Desired Settling Time (sec)	Desired Damping Ratio	Actual Settling Time (sec)	Max % Overshoot	Fuel Used
Adaptive	70	1.0	70	0	826
Non-adaptive	70	1.0	+100	55	722
Non-adaptive	11	2.5	70	4	958

The adaptive design clearly meets the desired settling time and desired overshoot requirements. The first nonadaptive design appears to use more fuel than the adaptive design, but it is very oscillatory and does not meet the desired settling time. A second nonadaptive design was generated by adjusting the desired settling time and damping ratio, which changes the parameters k and d in the control law developed in Chapter III, until the actual settling time approached

the original desired settling time of 70 seconds. This second nonadaptive design meets the required settling time but the response is more oscillatory and uses more fuel: as compared with the adaptive design. Figures 9, 10, and 11 further clarify the differences between the adaptive and nonadaptive designs.

The simulation plots of the quaternions or Euler parameters can be used to check for an eigenaxis rotation: which is defined by Ref. 9 as a straight line in any plot of the quaternions or Euler parameters. The first nonadaptive controller is clearly executing a noneigenaxis rotation. The second nonadaptive design controller and the adaptive controller are, almost exactly, executing an eigenaxis rotation. The adaptive controller is, therefore, the only design that:

1. minimizes the desired position angle overshoot;
2. meets the desired settling time;
3. minimizes the fuel used;
4. executes an eigenaxis rotation;
5. has the ability to react to different and changing moments of inertia.

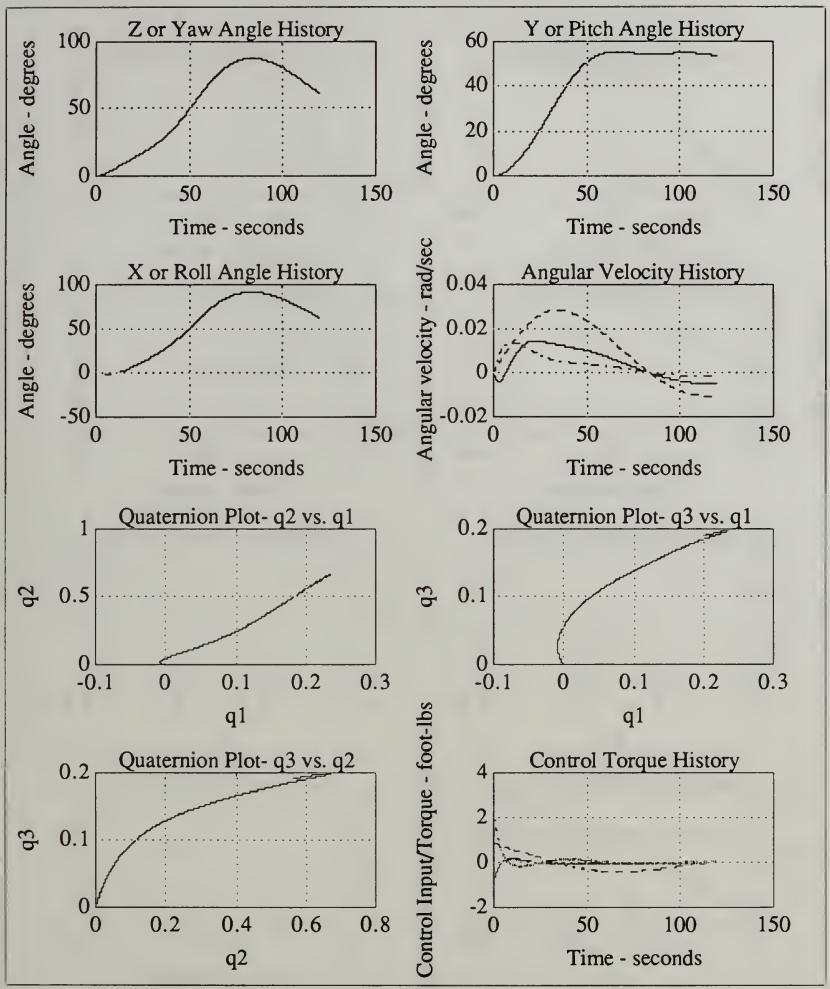

Figure 9. Simulation Results for Nonadaptive (Design 1) Slewing

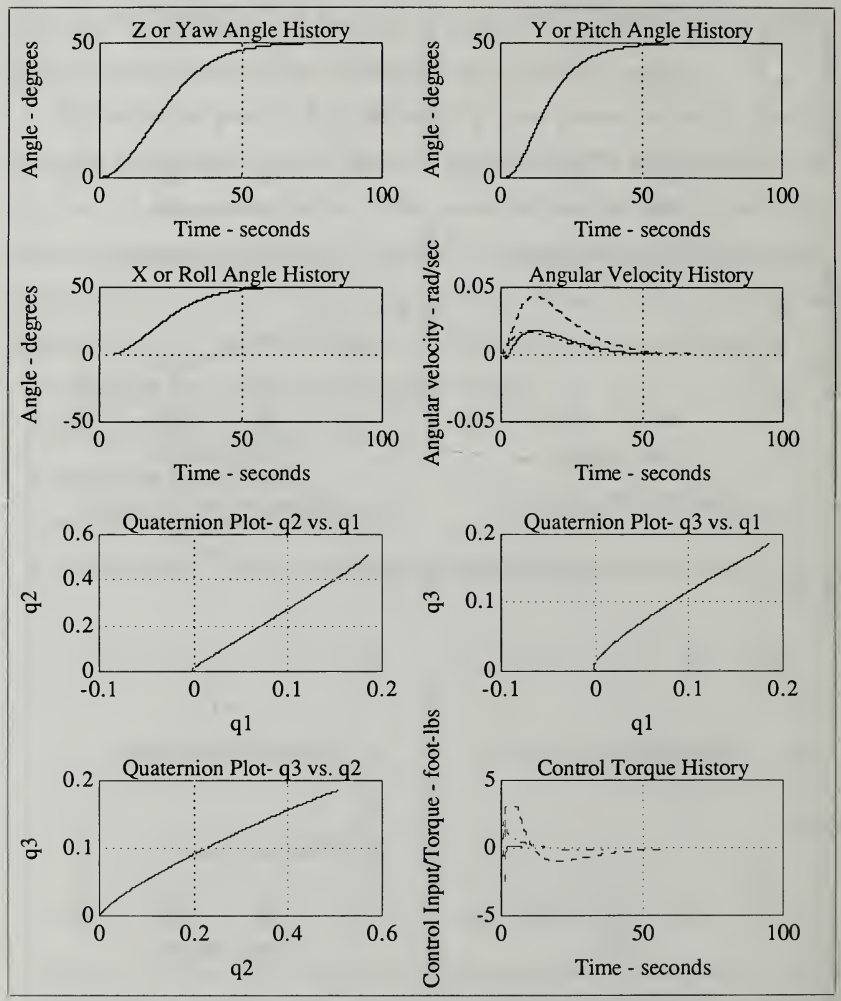

Figure 10. Simulation Results for Adaptive Slewing

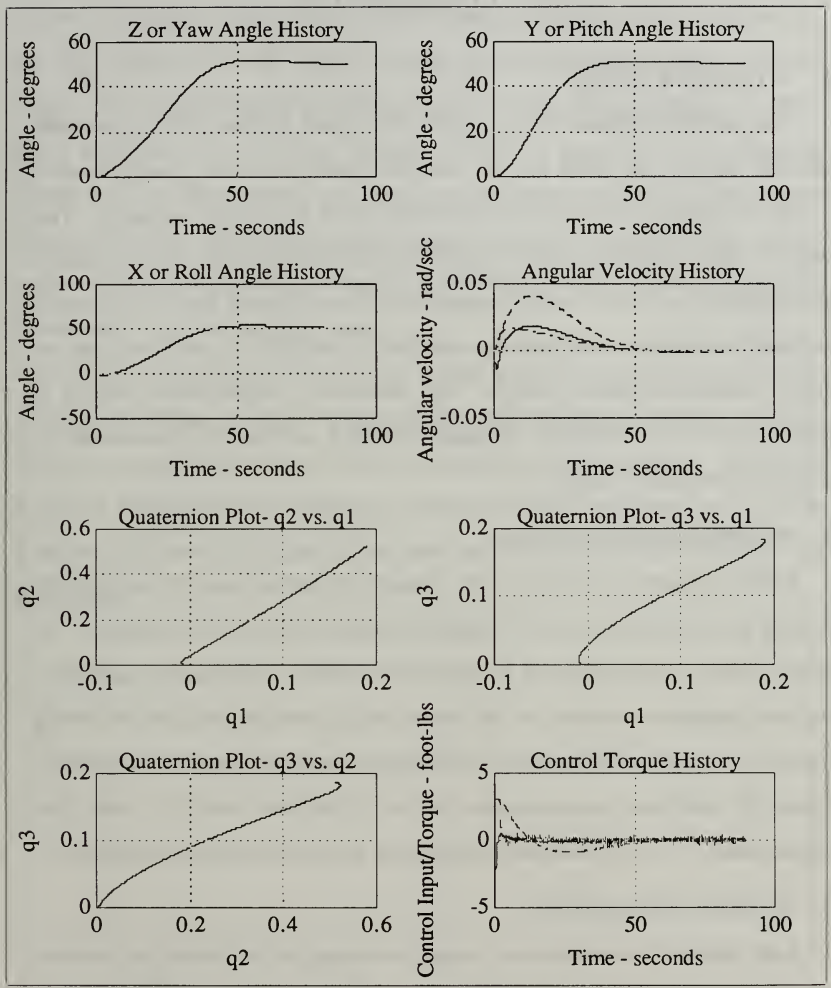

Figure 11. Simulation Results for Nonadaptive (Design 2) Slewing

VI. CONCLUSIONS

A. ATTITUDE HOLD

The adaptive attitude control system developed in this thesis to maintain attitude hold for the CER and an acquired target provided superior control during the capture of all previously defined worst case target scenarios. This adaptive linear quadratic regulator provided stable results with very reasonable settling times, and used a modest amount of fuel as compared with the previously designed nonadaptive minimum time control system [Ref. 3] and a nonadaptive linear quadratic regulator design. The additional computational burden of adaptive control is, therefore, compensated by a substantial improvement in control system performance.

B. SLEWING MANEUVERS

Fairly accurate knowledge of the moment of inertia tensor is essential for slewing maneuvers that are well damped, accomplished in a timely manner, use minimal fuel, and are executed as eigenaxis rotations. The adaptive quaternion feedback regulator developed in this thesis clearly provides this type of slewing maneuver control and outperforms nonadaptive quaternion feedback regulators. Again, the additional computational burden of adaptive control is more than compensated for by a substantial improvement in control system performance.

C. FUTURE RESEARCH

Both adaptive control system designs developed in this thesis are practical designs that will result in a more reliable and fuel efficient Crew/Equipment Retriever for the Space Station Freedom. These designs, however, can also be

applied to more general spacecraft attitude control problems. The adaptation scheme can easily be altered to account for spacecraft moments of inertia that are not only unknown but are subject to frequent and significant changes. In this respect, the adaptive linear quadratic regulator is an ideal candidate for large space structures of the future. Instead of limiting space structure design to shapes and load distributions that lead to quick approximations of the overall moment of inertia and minimize any changes, this adaptive control system would remove all these restrictions and become, in current science fiction terminology, an automatic attitude damping system.

Future research could examine alternative adaptation schemes and employ filtering techniques to obtain angular acceleration measurements from angular velocity measurements, since the latter are more generally available. Additional computer simulations could attempt to model the CER and target as a flexible structure and account for any target movement within the CER's capture net mechanism.

```
%       PROGRAM NAME:       CERSimNL1
%       PROGRAM AUTHOR:     Nicholas F. Russo
%       CALLED PROGRAMS:    actualmoi, CERLQR, guessmoi
%
%       DESCRIPTION:
%       The below program will simulate the CER Control System for Small
%   Angle Motion or Attitude Hold. The Control Law is a Steady State Linear
%   Quadratic Regulator.  However, it is an Adaptive Controller in that the
%   Inverse MOI of the CER + Target is unknown and potentially variable.  A
%   KALMAN Filter is used to estimate the Inverse MOI and then update the
%   feedback gains produced by the Linear Quadratic Regulator (LQR).  It is
%   assumed that the angular accelerations are available as measurements.
%
clear
%   Prompt user for first guess inverse MOI tensor
guessmoi;
%
% Define the initial state estimate for the Kalman Filter
est_x=[Iig(1,1);Iig(1,2);Iig(1,3);Iig(2,2);Iig(2,3);Iig(3,3)];
%
%  Define the Q and r parameters for the LQR
Q=diag([1,1,1,0,0,0]); % The Q matrix weights the system states
%
r=2.0e-5;  % This is used to weight the control inputs used
%  This is the for used  R=r*eye(3,3);
%
%       Define the actual inverse MOI tensor, as Ii.
%       This is placed into the MATLAB workspace by a separate
%       subroutine.
actualmoi;
%
T=75e-3;   % The sampling time.
B=[zeros(3,3);Ii]; %  The B matrix.
%
t(1)=1;  %Initial sampling index
% Initial conditions in degrees & degrees/sec  and in radians & radians/sec
x(:,1)=[2;2;2;0.2;0.2;0.2].*(pi/180)
%
y(:,1)=x(:,1);                  %  Initial measurements
%
```

```
% Initial Kalman Filter Parameters
phik=eye(6); % Define the phi matrix for the plant
%
Ew=(1e-8)*eye(6); % Plant Noise Covariance Matrix
V=1.0e-4;
Ev=V*eye(3); % Measurement Noise Covariance Matrix
%       Prompt user for Covariance Estimation Error
ques3=input('Specify the Covariance Estimation Error, P(0/0).  < 0 > The
Default. < 1 > The Identity Matrix. < 2 > A matrix of zeros.>>>');
%
if ques3==0,
oldP=[-0.0114 -1.14e-6 2.92e-5 2.166e-5 4.84e-7 2.95e-5
-1.14e-6 0.0001 -2.56e-7 -1.9e-7 -4.25e-9 -2.587e-7
2.92e-5 -2.56e-7 -0.0026 4.864e-6 1.087e-7 6.623e-6
1e-6 1e-6 1e-6 -.0019 1e-6 1e-6
1e-6 1e-6 1e-6 1e-6 -4.246e-5 1e-6
1e-6 1e-6 1e-6 1e-6 1e-6 -2.587e-3];
elseif ques3==1,
oldP=eye(6);
else,
oldP=zeros(6,6);
end
% This is P(0/0), the covariance matrix of the estimation error
%
% Prompt user for number of simulation steps
ques4=input('Enter the number of Simulation Steps. < 0 > The Default, which is
160.  Otherwise, enter your own value (within reason....unless you have all
day!!>>>');
if ques4==0,
NUM=160;
else,
NUM=ques4;
end
%
rand('normal');
%
rd=(1.4e-4)*rand(3,NUM);  %  Random Plant Disturbance
mn1=(1.745e-4)*rand(3,NUM);  % Random Measurement Noise
mn2=(5e-6)*rand(3,NUM);%
mn=[mn1;mn2];
%
%       Now, simulate the CER + Target
for N=1:NUM
```

```
%
wskew=[0 -x(6,N) x(5,N);x(6,N) 0 -x(4,N);-x(5,N) x(4,N) 0];
%       Skew symmetric matrix
a=-1*Ii*wskew*inv(Ii);
i5=tan(x(2,N))*sin(x(1,N));
i6=cos(x(1,N))*tan(x(2,N));
j5=cos(x(1,N));
j6=-sin(x(1,N));
k5=sin(x(1,N))/cos(x(2,N));
k6=cos(x(1,N))/cos(x(2,N));
A=[0 0 0 1 i5 i6;0 0 0 0 j5 j6;0 0 0 0 k5 k6;0 0 0 a(1,1) a(1,2) a(1,3);0 0 0 a(2,1)
a(2,2) a(2,3);0 0 0 a(3,1) a(3,2) a(3,3)];
%
% Now, discretize the state space equations
[phi,delu1]=c2d(A,B,T);
%
% Call the CERLQR function to determine the Steady State Feedback Gain
%  Matrix
K=CERLQR(Q,r,Iig);
%
t(N+1)=N+1;   % Increment time index
%
%
U(:,N)=-K*y(:,N);
% Now, apply the limits for the control torques
if abs(U(1,N)) > 4
U(1,N)=4*sign(U(1,N));
else
U(1,N)=U(1,N);
end
if abs(U(2,N)) > 3
U(2,N)=3*sign(U(2,N));
else
U(2,N)=U(2,N);
end
if abs(U(3,N)) > 3
U(3,N)=3*sign(U(3,N));
else
U(3,N)=U(3,N);
end
%
torque(:,N)=U(:,N)+rd(:,N);
%       The plant discrete state equations
```

```
x(:,N+1)=phi*x(:,N)+delu1*torque(:,N);
%              Measurement Equation
y(:,N+1)=x(:,N+1)+mn(:,N);
%
xd(:,N)=A*x(:,N)+B*U(:,N);
yka(:,N)=xd(4:6,N);   %  This measurement equation takes the angular velocities
%  and obtains angular accelerations (like an accelerometer ).
%
yk(:,N)=(y(4:6,N+1)-y(4:6,N))./T; % This equation obtains angular accelerations
%              by numerical differentiation.
%
%  Now, call the Kalman Filter  in order to estimate the
%  actual Inverse MOI and update the LQR Gain Matrix.
%
%
t1=U(1,N);
t2=U(2,N);
t3=U(3,N);
Ck=[t1 t2 t3 0 0 0;0 t1 0 t2 t3 0;0 0 t1 0 t2 t3];
%
%  Below are the Kalman Filter and gain equations
%
%  First, the Gain Equations
     newP=phik*oldP*phik' + Ew;
     G=newP*Ck'*inv(Ck*newP*Ck' + Ev);
     oldP=(eye(6) - G*Ck)*newP;
%
%
     est_xx=phik*est_x(:,N);
     est_y= Ck*est_xx;
     est_x(:,N+1)= est_xx + G*(yk(:,N)-est_y);
%
I11=est_x(1,N);
I12=est_x(2,N);
I13=est_x(3,N);
I22=est_x(4,N);
I23=est_x(5,N);
I33=est_x(6,N);
IEST=[I11 I12 I13;I12 I22 I23;I13 I23 I33];
%
%   Check the inverse MOI guess for physical reality.  Eigenvalues must be > 0.
eigcheck(1:3,N)=[eig(IEST)];
%
```

```
%  Test for physical reality  ( All eigenvalues of IEST >0)
checke(N)=1*sign(eigcheck(1,N))+1*sign(eigcheck(2,N))+1*sign(eigcheck(3,N))
;
%
if checke(N)==3,
%           Update the initial guess inverse MOI
Iig=IEST;
%           Otherwise, do not update the control law
else,
end
%
end                 % End of Simulation Loop
%
% Convert the time vector to actual time
t=t.*T;
% create a special time vector for the control input plot
tU=0:N-1;
tU=tU.*T;
%           Calculate the total fuel used (actually this is proportional to it)
FUELUSED=sum(sum(abs(U)))
%
%           End of Program
```

```
%     PROGRAM NAME:              guessmoi
%     PROGRAM AUTHOR:            Nicholas F. Russo
%     CALLED PROGRAMS:   none
%
%     DESCRIPTION:
%     The below program will prompt user for first guess inverse MOI tensor.
input('Welcome to the Non-Linear Simulation of the CER and Target. Please
choose a first guess inverse MOI tensor for the LQR. <0 >  The Default.
MAN+MMU in Net Center with CER Frame of Reference.< 1 > The CER alone.
< 2 >  Case Two from Hansen Thesis.  *Any other number will use an average
of all the inverse MOI tensors>>>');
s=ans;
if s==2,
Iig=[0.0114  -0.0001  0.00256;-0.0001  0.0019  4.246e-5;0.00256  4.246e-5
0.002587];
%
elseif s==1,
Iig=inv([39.6 0 0;0 55 0;0 0 55]);
elseif s==0,
Iig=[0.0138 0 0.003;0 0.0018 0;0.003 0 0.0026]
```

```
%
else,
Iig=inv([143.8 -33.6 -175.2;-33.6 646 -10.5;-175.2 -10.5 562])
end
%
Ig=inv(Iig)
%       This is the guess MOI.
%
%       End of Program
```

```
%       PROGRAM NAME:       actualmoi
%       PROGRAM AUTHOR:     Nicholas F. Russo
%       CALLED PROGRAMS:    none
%
%       DESCRIPTION:
%       This program will question the user for a CER + target
%       moment of inertia or its inverse.
%
caseN=input('Define the actual Moment of Inertia by case number>>');
if caseN==1,
MOI=[39.6 0 0;0 55 0;0 0 55];
%       This is the CER alone.
%
elseif caseN==2,
MOI=[112.9 2.4 -111.9;2.4 534.9 6.4;-111.9 6.4 497.6];
%       MAN + MMU
%
elseif caseN==3,
MOI=[69.3 0 -178.2;0 1153.8 0;-178.2 0 1124.1];
%
elseif caseN==4,
MOI=[98.9 -110.5 -110.5;-110.5 496.1 -29.7;-110.5 -29.7 496.1];
%
elseif caseN==5,
MOI=[369.5 0 -368.4;0 796.4 0;-368.4 0 466.4];
%
elseif caseN==6,
MOI=[172.7 -93.7 -282.3;-93.7 839.7 -39.8;-282.3 -39.8 732.9];
%
else,
end
Ii=inv(MOI);
%
```

```
%      End of Program
```

```
%      PROGRAM NAME:        CERLQR
%      PROGRAM AUTHOR:      Nicholas F. Russo
%      CALLED PROGRAMS:     dlqr
%
%      DESCRIPTION:
%  The below subroutine will compute the Steady State Linear
%  Quadratic Regulator Control Gains for the CER using a Linear
%  State Space Model. It will receive updates on the inverse
%  Moment of Inertia (MOI) Tensor from another subroutine
%  that will estimate this matrix using a Kalman Filter.
%
%      function K= CERLQR(Q,r,Ii)
%   Here are the input parameters:
%   Q is the weighting matrix for the states
%   r is the scalar used to weight the control inputs
%     Ii is the inverse MOI Tensor
%
function K=CERLQR(Q,r,Ii);
%
R=r*eye(3,3);
%
%  Now, define the state space equations and discretize.
T=75e-3;   % The sampling time.
A=[0 0 0 1 0 0;0 0 0 0 1 0;0 0 0 0 0 1;0 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0];
% The above is the A matrix.
%
B=[zeros(3,3);Ii]; %  The B matrix.
%  Now, discretize the state space equations
[phi,delu1]=c2d(A,B,T);
%
%      Now, call the function dlqr to calculate the feedback gains
%
[K,S]=dlqr(phi,delu1,Q,R);
%  END of the function
```

```
%      PROGRAM NAME:        replotSA
%      PROGRAM AUTHOR:      Nicholas F. Russo
%      CALLED PROGRAMS:     subplot, plot
%
%      DESCRIPTION:
%      Plotting Program for Small Angle Motion
```

```
clg;
subplot(221);
plot(t,x(1,:),t,x(2,:),'--',t,x(3,:),'-.'),grid
xlabel('Time - seconds'),ylabel('Position - radians')
title('Position Angle Histories')
%
subplot(222);
plot(t,x(4,:),t,x(5,:),'--',t,x(6,:),'-.'),grid
xlabel('Time - seconds'),ylabel('Angular velocity - rad/sec')
title('Angular Velocity Histories')
%
subplot(223);
plot(tU,U(1,:),tU,U(2,:),'--',tU,U(3,:),'-.'),grid
xlabel('Time - seconds'),ylabel('Control Torque - foot-lbs')
title('Control Torque Histories')
%
subplot(224);
%       Plot fuel usage plot
plot(tU,sum(abs(U))),grid,title('Fuel Usage')
xlabel('Time - seconds'),ylabel('Magnitude');
pause;
%  Prompt user to see if additional plots/information are desired.

ques5=input('Would you like to see the final estimate of the Inverse Moment of
Inertia Matrix?? If the answer is yes then type 0, otherwise type  1 .>>>');
if ques5==0,
Ii
pause;
Iig
pause;
clg;
subplot(221);
plot(t,est_x(1,:)),grid
title('      Inverse Moment of Inertia Estimate- I11')
xlabel('Time - seconds')
subplot(222);
plot(t,est_x(2,:)),grid,title('I12')
xlabel('Time - seconds')
subplot(223);
plot(t,est_x(3,:)),grid,title('I13')
xlabel('Time - seconds')
subplot(224);
plot(t,est_x(4,:)),grid,title('I22')
```

```
xlabel('Time - seconds'),pause;
clg;
subplot(221);
plot(t,est_x(5,:)),grid,xlabel('Time - seconds')
title('Inverse Moment of Inertia Estimate - I23')
subplot(222);
plot(t,est_x(6,:)),grid,xlabel('Time - seconds')
title('I33'),pause;
clg;
plot(eigcheck(1,:)),grid,title('First Eigenvalue of the Estimated Inverse MOI
Tensor'),pause
plot(eigcheck(2,:)),grid,title('Second Eigenvalue'),pause
plot(eigcheck(3,:)),grid,title('Third Eigenvalue'),pause
%
else,
end
%       End of Program
```

```
%       PROGRAM NAME:           CERSlewK1
%       PROGRAM AUTHOR:         Nicholas F. Russo
%       CALLED PROGRAMS:        guessmoi, actualmoi, EULERa2p,
%                               EULERp2a321
%
%       DESCRIPTION:
%       The below program will simulate the CER Control System for Large
%       Angle (Slewing) Motion.  The Control Law is a Quaternion Feedback
%       Regulator.  However, it is an Adaptive Controller in that the MOI
%       of the CER + Target is unknown and potentially variable.  A KALMAN
%       Filter is used to estimate the  MOI and then update the feedback gains
%       produced by the Quaternion Feedback Regulator.  This KALMAN Filter
%       assumes that angular acceleration measurements are available.
clear
%   Prompt user for first guess inverse MOI tensor
%
guessmoi;
% Define the initial state estimate for the Kalman Filter
est_x=[Ig(1,1);Ig(1,2);Ig(1,3);Ig(2,2);Ig(2,3);Ig(3,3)];
%
%       Define the actual MOI tensor.
%       This is placed into the MATLAB workspace by a separate subroutine.
actualmoi;
%
T=75e-3;   % The sampling time.
```

```
t(1)=1;  %Initial sampling index
%  Initial orientation
anginit=[0 0 0];
%
xEULER(1:3,1)=[anginit(1);anginit(2);anginit(3)];
%       Prompt the user for the desired orientation
angc=input('Specify the final or desired orientation in terms of 3-2-1 Euler
Angles, in degrees.  Enter as follows, [120 120 120]>>>>>');
%
%       Now, convert each of these orientations to Euler Parameters
%       Initial Conditions
binit=EULERa2p(2,0,anginit(1),anginit(2),anginit(3));
beta(:,1)=[binit(1);binit(2);binit(3);binit(4)];
w(:,1)=[0;0;0];
%
bcmd= EULERa2p(2,0,angc(1),angc(2),angc(3));
%       Compute the commanded quaternion matrix
b0c=bcmd(1);
b1c=bcmd(2);
b2c=bcmd(3);
b3c=bcmd(4);
BCMD=[b0c b1c b2c b3c;-b1c b0c b3c -b2c;-b2c -b3c b0c b1c;-b3c b2c -b1c
b0c];
%
%       Prompt the user for the values used to compute the K and D weighting
%       matrices used in the Quaternion Feedback Regulator.
settle=input('Please enter the desired Settling Time in seconds>>>>');
%
runtime=input('Enter desired simulation run time (sec)>>>>');
actual=runtime/T;
NUM=actual+0.2*actual;
NUM=round(NUM);
%
%  The two weighting matrices, K and D will now be calculated as follows:
zeta=1.0
%
omegaN=8/(zeta*settle);
dscalar=2*zeta*omegaN;
kscalar=2*omegaN^2;
%
rand('normal');
%
rd=(1.4e-4)*rand(3,NUM);  %  Random Plant Disturbance
```

```
mn1=((1.745e-4)/3)*rand(4,NUM);  % Random Measurement Noise
mn2=((5e-6)/3)*rand(3,NUM);%
%
%  Initial Kalman Filter Parameters
phik=eye(6); % Define the phi matrix for the plant
%       Prompt user for Kalman Filter Design Plant Noise
Wnoise=input('Specify the scalar multiplier for the Kalman Filter Plant Noise
Matrix.  < 0 >  Default Value (TRUST ME).  < 1 > No NOISE.  < 2 > Choose
your own value.>>>>>');
if Wnoise==0,
W=1e-8;  % The scalar multiplier of the below matrix
%
elseif Wnoise==1,
W=0;
else,
W=input('Specify your own value.>>>>>');
end
Ew=W*eye(6);  % Plant Noise Covariance Matrix
V=1.0e-04;
Ev=V*eye(3);  % Measurement Noise Covariance Matrix
%       Prompt user again, for Covariance Estimation Error
ques3=input('Specify the Covariance Estimation Error, P(0/0).  < 0 > The
Default.  < 1 >  The Identity Matrix.  < 2 > A matrix of zeros.>>>');
%
if ques3==0,
oldP=inv([-0.0114 -1.14e-6 2.92e-5 2.166e-5 4.84e-7 2.95e-5
-1.14e-6 0.0001 -2.56e-7 -1.9e-7 -4.25e-9 -2.587e-7
2.92e-5 -2.56e-7 -0.0026 4.864e-6 1.087e-7 6.623e-6
1e-6 1e-6 1e-6 -.0019 1e-6 1e-6
1e-6 1e-6 1e-6 1e-6 -4.246e-5 1e-6
1e-6 1e-6 1e-6 1e-6 1e-6 -2.587e-3]);
elseif ques3==1,
oldP=eye(6);
else,
oldP=zeros(6,6);
end
%  This is P(0/0), the covariance matrix of the estimation error
%
tclock0=clock;
%       Initialize omega with noise
wN(:,1)=w(:,1);
%       Now, simulate the CER + Target
for N=1:NUM
```

```
%
wN(:,N+1)=w(:,N)+mn2(:,N);  % angular velocity plus measurement noise.
%
betaN(:,N)=beta(:,N)+mn1(:,N);  %  Position measurements plus noise.
wskewN=[0 -wN(3,N) wN(2,N);wN(3,N) 0 -wN(1,N);-wN(2,N) wN(1,N) 0];
%
wskew=[0 -w(3,N) w(2,N);w(3,N) 0 -w(1,N);-w(2,N) w(1,N) 0];
%
Gskew=.5*[0 -w(1,N) -w(2,N) -w(3,N);w(1,N) 0  w(3,N) -w(2,N);w(2,N)
-w(3,N) 0 w(1,N);w(3,N) w(2,N) -w(1,N) 0];
%
t(N+1)=N+1;   % Increment time index
%
%
%       Define the error quaternion for the feedback equation
be=BCMD*betaN(:,N);
qe=[be(2);be(3);be(4)];
%      Define the weighting matrices K and D.  Note that they are updated by the
%      Kalman Filter's Estimate of the MOI.
K=kscalar*Ig;
Dq=dscalar*Ig;
U(:,N)=wskewN*Ig*wN(:,N) - Dq*wN(:,N) - K*qe;
%  Now, apply the limits for the control torques
if abs(U(1,N)) > 4
U(1,N)=4*sign(U(1,N));
else
U(1,N)=U(1,N);
end
if abs(U(2,N)) > 3
U(2,N)=3*sign(U(2,N));
else
U(2,N)=U(2,N);
end
if abs(U(3,N)) > 3
U(3,N)=3*sign(U(3,N));
else
U(3,N)=U(3,N);
end
%
torque(:,N)=U(:,N)+rd(:,N);  %  Apply the random disturbance torques
%
%              Discrete Nonlinear Kinematic and Dynamic Equations
w(:,N+1)=w(:,N)+T*(Ii*torque(:,N)-Ii*wskew*MOI*w(:,N));
```

```
beta(:,N+1)=beta(:,N)+T*Gskew*beta(:,N);
%   The real angular acceleration
wdotreal(:,N)=Ii*torque(:,N)-Ii*wskew*MOI*w(:,N);
%   Angular acceleration from numerical differentiation
wdot(:,N)=(wN(:,N+1)-wN(:,N))./T;
%
%  Now, call the Kalman Filter  in order to estimate the
%  actual MOI and update the Control Law.
%
%
Ck=[wdot(1,N) (wdot(2,N)-wN(1,N)*wN(3,N)) (wN(1,N)*wN(2,N)+wdot(3,N))
(-wN(2,N)*wN(3,N)) (wN(2,N)^2-wN(3,N)^2) (wN(2,N)*wN(3,N))
wN(1,N)*wN(3,N) (wdot(1,N)+wN(2,N)*wN(3,N)) (wN(3,N)^2-wN(1,N)^2)
wdot(2,N) (wdot(3,N)-wN(1,N)*wN(2,N)) -wN(1,N)*wN(3,N)
-wN(2,N)*wN(1,N) (wN(1,N)^2-wN(2,N)^2) (wdot(1,N)-wN(2,N)*wN(3,N))
wN(1,N)*wN(2,N) (wdot(2,N)+wN(1,N)*wN(3,N)) wdot(3,N)];
%
%  Below are the Kalman Filter and gain equations
%
%  First, the Gain Equations
     newP=phik*oldP*phik' + Ew;
     G=newP*Ck'*inv(Ck*newP*Ck' + Ev);
     oldP=(eye(6) - G*Ck)*newP;
%
     est_xx=phik*est_x(:,N);
     est_y= Ck*est_xx;
     est_x(:,N+1)= est_xx + G*(torque(:,N)-est_y);
%
I11=est_x(1,N);
I12=est_x(2,N);
I13=est_x(3,N);
I22=est_x(4,N);
I23=est_x(5,N);
I33=est_x(6,N);
IEST=[I11 I12 I13;I12 I22 I23;I13 I23 I33];
%
%   Check the inverse MOI guess for physical reality.  Eigenvalues must be > 0.
eigcheck(1:3,N)=[eig(IEST)];
%
checke(N)=1*sign(eigcheck(1,N))+1*sign(eigcheck(2,N))+1*sign(eigcheck(3,N))
;
%
if checke(N)==3,
```

```
%               Update the initial guess MOI
Ig=IEST;
%               Otherwise, do not update the control law
else,
end
%
%
%               Convert the Euler Parameters to Euler Angles
xEULER(:,N+1)=EULERp2a321(beta(:,N+1));
end                             %  End of Simulation Loop
%
%  Convert the time vector to actual time
t=t.*T;
%  create a special time vector for the control input plot
tU=0:N-1;
tU=tU.*T;
etime(clock,tclock0)/60
%               Calculate the total fuel used (actually this is proportional to it)
FUELUSED=sum(sum(abs(U)))
%
%               End of CER Slewing Simulation Program
```

```
%       PROGRAM NAME:       replot
%       PROGRAM AUTHOR:     Nicholas F. Russo
%       CALLED PROGRAMS:    plot, subplot
%
%       DESCRIPTION:
%       Plotting program for previously calculated data
clg;
subplot(221);
plot(t,xEULER(1,:)),grid
xlabel('Time - seconds'),ylabel('Angle - degrees')
title('Z or Yaw Angle History')
%
subplot(222);
plot(t,xEULER(2,:)),grid
xlabel('Time - seconds'),ylabel('Angle - degrees')
title('Y or Pitch Angle History')
%
subplot(223);
plot(t,xEULER(3,:)),grid
xlabel('Time - seconds'),ylabel('Angle - degrees')
title('X or Roll Angle History')
```

```
%
subplot(224);
plot(t,w(1,:),t,w(2,:),'--',t,w(3,:),'-.'),grid
xlabel('Time - seconds'),ylabel('Angular velocity - rad/sec')
title('Angular Velocity History')
pause;
clg;
%
subplot(221);
plot(beta(2,:),beta(3,:)),grid
title('Quaternion Plot- q2 vs. q1'),xlabel('q1'),ylabel('q2')
%
subplot(222);
plot(beta(2,:),beta(4,:)),grid
title('Quaternion Plot- q3 vs. q1'),xlabel('q1'),ylabel('q3')
%
subplot(223);
plot(beta(3,:),beta(4,:)),grid
title('Quaternion Plot- q3 vs. q2'),xlabel('q2'),ylabel('q3')
%
subplot(224);
plot(tU,U(1,:),tU,U(2,:),'--',tU,U(3,:),'-.'),grid
title('Control Torque History')
xlabel('Time - seconds'),ylabel('Control Input/Torque - foot-lbs')
pause;
clg;
%
%  Prompt user to see if additional plots/information are desired.
ques5=input('Would you like to see the final estimate of the Inverse Moment of
Inertia Matrix??  < 0 > YES.     < 1 > NO >>>');
if ques5==0,
MOI
pause;
MOIestimate=IEST
pause;
invMOI=Ii
pause;
invMOIestimate=inv(IEST)
pause;
MOIeigenvalues=eig(MOI)
pause;
MOIesteigenvalues=eig(MOIestimate)
pause;
```

```
subplot(221);
plot(t,est_x(1,:)),grid
title('Moment of Inertia Estimate - I11')
xlabel('Time - seconds')
subplot(222);
plot(t,est_x(2,:)),grid,title('I12')
xlabel('Time - seconds')
subplot(223);
plot(t,est_x(3,:)),grid,title('I13')
xlabel('Time - seconds')
subplot(224);
plot(t,est_x(4,:)),grid,title('I22')
xlabel('Time - seconds'),pause;
clg;
subplot(221);
plot(t,est_x(5,:)),grid,xlabel('Time - seconds')
title('Moment of Inertia Estimate - I23')
subplot(222);
plot(t,est_x(6,:)),grid,xlabel('Time - seconds')
title('I33')
subplot(223);
%       Plot fuel usage plot
plot(sum(abs(U))),grid,title('Fuel Usage'),pause;
%
clg;
%
plot(eigcheck(1,:)),grid,title('First Eigenvalue of the Estimated MOI
Tensor'),pause
plot(eigcheck(2,:)),grid,title('Second Eigenvalue'),pause
plot(eigcheck(3,:)),grid,title('Third Eigenvalue'),pause
%
else,
end
%
%       End of Program
```

```
%       PROGRAM NAME:       EULERa2p
%       PROGRAM AUTHOR:     Nicholas F. Russo
%       CALLED PROGRAMS:    none
%
%       DESCRIPTION:
%       The below function will compute the four Euler parameters
%       given a sequence of three Euler Angles.  A 3-2-1 or a 3-1-2
```

```
%      rotation sequence is allowed.
%
%  function E=EULERa2p(e,r,a1,a2,a3);
%
%              Here are the required input arguments:
%  Is the sequence 3-2-1 (e=2) or 3-1-2 (e=1)?
%              Are the three angles in radians ? (Yes: r=1, No: r=0)
%              The three Euler Angles (a1,a2,a3)
%
%     A vector is returned that contains b0,b1,b2,b3.
function E=EULERa2p(e,r,a1,a2,a3);
%
%              Convert the angles to radians if necessary
if r==0
%
a1=a1*pi/180;
a2=a2*pi/180;
a3=a3*pi/180;
%
else
end
%              Define the R matrices
R0=[1 0 0 0;0 1 0 0;0 0 1 0;0 0 0 1];
R1=[0 -1 0 0;1 0 0 0;0 0 0 1;0 0 -1 0];
R2=[0 0 -1 0;0 0 0 -1;1 0 0 0;0 1 0 0];
R3=[0 0 0 -1;0 0 1 0;0 -1 0 0;1 0 0 0];
%
%      Test to see which rotation sequence is desired and proceed.
%
if e==1  %  This is a 3-1-2 sequence
R312=(cos(a3/2)*R0
+sin(a3/2)*R2)*(cos(a2/2)*R0+sin(a2/2)*R1)*(cos(a1/2)*R0+sin(a1/2)*R3);
E=R312*[1;0;0;0];
else
%  This is a 3-2-1 sequence.
R321=(cos(a3/2)*R0+sin(a3/2)*R1)*(cos(a2/2)*R0+sin(a2/2)*R2)*(cos(a1/2)*R0
+sin(a1/2)*R3);
E=R321*[1;0;0;0];
end
%
%  End of function
```

```
%      PROGRAM NAME:          EULERp2a321
```

```
%      PROGRAM AUTHOR:     Nicholas F. Russo
%      CALLED PROGRAMS:    none
%
%      DESCRIPTION:
%      The below function will calculate a sequence of three Euler Angles
%  from given Euler Parameters. A 3-2-1        rotation sequence is
%  allowed.
%
%  function E=EULERp2a321(bvector)
%
%              Here is the required input argument:
%              The 4 Euler Parameters:  b0,b1,b2,b3
%              Enter these parameters as a vector.  For example,
%              bvector=[b0;b1;b2;b3]
%     A vector is returned that contains the three angles in degrees.
function E=EULERp2a321(bvector);
%
b0=bvector(1);
b1=bvector(2);
b2=bvector(3);
b3=bvector(4);
% First, compute the Direction Cosine Matrix (DCM) from the Euler
Parameters.
Cp=[b0^2+b1^2-b2^2-b3^2  2*(b1*b2+b0*b3)  2*(b1*b3-b0*b2);2*(b1*b2-
b0*b3) b0^2-b1^2+b2^2-b3^2 2*(b2*b3+b0*b1);2*(b1*b3+b0*b2) 2*(b2*b3-
b0*b1) b0^2-b1^2-b2^2+b3^2];
%
%              Calculate the middle angle, beta
a2=atan2(-Cp(1,3),((Cp(1,1))^2+(Cp(1,2))^2)^.5);
%        Test this angle and calculate the other angles
a2d=a2*(180/pi);
a2d=round(a2d);
if a2d==90,
a1=0;%  This is the angle alpha or the rotation about the z-axis
a3=atan2(Cp(2,1),Cp(2,2));
%
elseif a2d==-90,
a1=0;
a3=-atan2(Cp(2,1),Cp(2,2));
%
else,
a1=atan2((Cp(1,2)/cos(a2)),(Cp(1,1)/cos(a2)));
a3=atan2((Cp(2,3)/cos(a2)),(Cp(3,3)/cos(a2)));
```

```
end
%
E=[a1;a2;a3];
E=(180/pi)*E;
%
%  End of function
```

```
%       PROGRAM NAME:        Inew
%       PROGRAM AUTHOR:      Nicholas F. Russo
%       CALLED PROGRAMS:     none
%
%       DESCRIPTION:
%       The below function will calculate the new Moment of Inertia (MOI)
%     Tensor for the CER plus the Target.
%       The Target's MOI can be in the CER ref frame or its own;
%    as long as a Direction Cosine Matrix (DCM) is available
%   function I=Inew(Icer,Itar,M1,M2,r2,ref,dcm);
%
%         Here are the required input arguments:
%         MOI of CER (Icer)
%     MOI of Target (Itar)
%     ref  (If it equals 0 then Itar is in the CER ref frame)
%       DCM from the Target ref frame to the CER ref frame (dcm)
%       Mass of the CER (M1)
%    Mass of the Target (M2)
%    The vector between the Center of Mass of the CER
%    and the target mass in Cartesian Coord.   (r2)
%
function  I=Inew(Icer,Itar,M1,M2,r2,ref,dcm);
%         First, test to see if the Target MOI is in the CER ref frame
if ref==0
Itar = Itar;
else
Itar= dcm*Itar*dcm';
end
%
%     Define the elements of the R2 Matrix
R2=[0 -r2(3) r2(2);r2(3) 0 -r2(1);-r2(2) r2(1) 0];
MF=M1*M2/(M1+M2); %  The Mass Factor multiplying R2
%
I= Icer + Itar  - MF*R2*R2;
%   End of Function
```

```
%      PROGRAM NAME:        DCM
%      PROGRAM AUTHOR:      Nicholas F. Russo
%      CALLED PROGRAMS:     none
%
%      DESCRIPTION:
%  The below function will calculate the Direction Cosine Matrix
%      (DCM) for a 3 Euler Angle rotation sequence.  A 3-2-1 (Yaw,
%  Pitch, Roll) or a 3-1-2 (Yaw, Roll, Pitch) is allowed.
%
%  function D=DCM(e,r,a1,a2,a3);
%
%  Here are the required input arguments:
%       Is the sequence 3-2-1 (e=2) or 3-1-2 (e=1) ?
%             The three Euler Angles (a1,a2,a3), in radians.
%             Are the Euler Angles in radians (Yes: r=1, No: r=0)
function D=DCM(e,r,a1,a2,a3);
%
%   Convert the angles to radians if necessary
if r==0
a1=a1*pi/180;
a2=a2*pi/180;
a3=a3*pi/180;
else
end
%  First, set up the individual rotation matrices.
C3=[cos(a1) sin(a1) 0;-sin(a1) cos(a1) 0;0 0 1];
%  This corresponds to a rotation about the # 3 axis
%
C2a=[cos(a2) 0 -sin(a2);0 1 0;sin(a2) 0 cos(a2)] ;% # 2 axis
%
C2b=[cos(a3) 0 -sin(a3);0 1 0;sin(a3) 0 cos(a3)];  % # 2 axis
%
C1a=[1 0 0;0 cos(a3) sin(a3);0 -sin(a3) cos(a3)];  % # 1 axis
%
C1b=[1 0 0;0 cos(a2) sin(a2);0 -sin(a2) cos(a2)];   % # 1 axis
%
%
%  Test to see which rotation sequence is desired and proceed.
%
if e==1
D= C2b*C1b*C3;
else
D= C1a*C2a*C3;
```

```
D= C1a*C2a*C3;
end
%
%  End of this function
```

LIST OF REFERENCES

1. *Space Station Projects Office Work Package 2 Request for Proposal* (9-BF2-51-7-1P-P01), National Aeronautics and Space Administration, Johnson Space Center, May 1987.

2. *Best and Final Offer; Responses to Questions and RFP Amendment 7 Book 2*, McDonnell Douglas Astronautics Company-Space Station Division, 25 September 1987.

3. Hansen, Daniel L., *Modeling, Simulation, and Analyses of Attitude Control for the Crew/Equipment Retriever Proposed for Space Station*, Master's Thesis, Naval Postgraduate School, Monterey, California, March 1989.

4. Wertz, James R., *Spacecraft Attitude Determination and Control*, D. Reidel Publishing Company, Holland, 1986.

5. Junkins, John L., Turner, James D., *Optimal Spacecraft Rotational Maneuvers*, Elsevier, Holland, 1989.

6. Byers, Robert M., *Time-Optimal Spacecraft Reorientation: A Quasi Closed-Form Control Law*, Ph.D. Dissertation, Texas A&M, College Station, Texas, May 1991.

7. Franklin, Gene F., Powell, J. David, Emami-Naemi, Abbas, *Feedback Control of Dynamic Systems*, 2nd ed., Addison-Wesley, 1991.

8. Friedland, Bernard, *Control System Design: An Introduction to State Space Methods*, McGraw-Hill, 1986.

9. Wie, B., Weiss, H., Arapostathis, A., "Quaternion Feedback Regulator for Spacecraft Eigenaxis Rotations," *Journal of Guidance, Control, and Dynamics*, v. 12, pp. 375-380, May-June 1989.

10. Slotine, Jean-Jacques E., Li, Weiping, *Applied Nonlinear Control*, Prentice Hall, 1991.

11. Burl, Jefferey B., "Linear Optimal Estimation and Control Notes," Naval Postgraduate School, Monterey, California, September 1991.

12. Kaplan, Marshall H., *Modern Spacecraft Dynamics & Control*, John Wiley & Sons, 1976.

13. Interview between Dr. John L. Junkins, visiting professor from Texas A&M, and the author, 20 April 1992.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5100	2
3.	Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4.	Prof. Jeff B. Burl, Code EC/Bl Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	2
5.	Prof. Roberto Cristi, Code EC/Cx Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
6.	Prof. H. A. Titus, Code EC/Ts Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
7.	Dr. John L. Junkins, Code AA/Ju Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, California 93943-5000	1
8.	Prof. Brij Agrawal, Code AA/Ag Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, California 93943-5000	1

9. Commandant (G-PIM-2/O) 1
U.S. Coast Guard
2100 2nd Street, S.W.
Washington, D.C. 20593-0001

10. Commandant (G-TPR-2) 1
U.S. Coast Guard
2100 2nd Street, S.W.
Washington, D.C. 20593-0001

11. Commanding Officer (ng) 2
U. S. Coast Guard Electronics Engineering Center
P. O. Box 60
Wildwood, New Jersey 08260-0060
Attn: LT Nicholas F. Russo

Thesis
R919 Russo
c.1 The design of an
adaptive attitude
control system.